An application programmable interface (API) configured to present a selection of video games having interactive gaming content responsive to a variety of physical motion of the exercise equipment that is powered by the equipment user and designed to encourage ongoing participation by the user in various physical activity scenarios. The API is configured to merge exercise equipment with video gaming that is inter-actively compatible across multiple exerciser hardware configurations, across multiple gaming system equipment configurations, and across independent display monitors. Gaming content is designed to interactively encourage continued and varying exercising by associating the motions of the exercise equipment with changes in audio visual content presented on the monitor or by other gaming system configurations.
Fig. 8
The following diagram illustrates the structure and association of the various Exercise hardware and software components. The various object oriented classes (HRG_C...) shown here in three groups, will be combined into one or several APIs. As we go from Group 1 to 3, the level of abstraction increases. Group 2 currently shows representations of bikes and treadmills. This can be expanded to include steppers, elliptical trainers etc. Preferably as new genres applications and games are created, the software tools (new classes) can be added to Group 3 - hence the blank boxes. Preferably, Group 3 is where most of the development can be done.

Fig. 9
Passes session results. Session results are ordered by original set descriptor and contain:
- original set descriptor,
- actual duration,
- max HR,
- min HR,
- average intensity.
Optional parameter is:
- # HR samples,
- time between samples,
- HR samples.

Game calls Tick() to give runtime cycles to core.
Input: float GameIntensity
Output: int currentHR float currentIntensity float currentSpeed

Activity initializes Core with ActivityStyle.
Core responds with (device capability, user, input:...)

iExercise calls SetChangeCallback()
Input: this set descriptor, next set descriptor
Output: SetResult

iExercise calls SetChangeRequest()
Input: [next | warmdown | quit]
Output: <none>

Initialization:
Passes setup parameters: TBD
Gets device capability:
- Min intensity
- Max intensity
- Speed steps
- Non-linearity profile

Runtime:
Sets intensity
Gets speed, HR

Initializes:
Passes session results, session results are ordered by original set descriptor and contain:
- original set descriptor,
- actual duration,
- max HR,
- min HR,
- average intensity.
Optional parameter is:
- # HR samples,
- time between samples,
- HR samples.
Fig. 11

iExercise Training Profile Setup

Default, user, coach or Server Profile

Session results: Passes final time, duration: min, max, HR, average.

Optional parameter: # HR samples, time between samples, HR samples.

Game calls: Tick() to give runtime cycles to core.

Input: float GameIntensity
Output: int currentHR, float currentIntensity, float currentSpeed

iExercise Runtime Core

Initialization: Ticks setup parameters: TBD

Get device capability: Min intensity, Max intensity, intensity steps, Runtime cycles

Non-linearity profile

iExercise Universal Device Driver

Initialization: Passes setup parameters: TBD

Get device capability:

Min intensity
Max intensity
Intensity steps
Non-linearity profile

iExercise Compliant Device

Run-time transactions

iExercise Compliant Application

Input: GameDeviceInput
Output: Game Device Output

Sights and sounds from game or other activity environment.

Happy User

Physical Exercise

Exercise Profile Initialization
User selects music: warmup, warmdown, intensity, recovery. Indicates whether music should control set length or whether music should be faded in/out as required.

Initialization:
- Passes setup parameters: TBD
- Gets device capability:
  - Min intensity
  - Max intensity
  - Intensity steps
  - Non-linearity profile

Runtime:
- Gets speed, HR

Activity initializes Core with ActivityStyle. Core responds with device capability, user, and intensity data and other "session" data.

iExercise calls SetChangeCallback() input:
- The set descriptor, next set descriptor
Output: SetResult

Game calls SetChangeRequest() input:
- next | warmup | quit
Output: <none>

Fig. 12
IEExercise uses USB to connect fitness devices to Xbox and PS2 game consoles and PCs

Fig. 15
<table>
<thead>
<tr>
<th>Market Players</th>
<th>Standard Interface with 3rd Party Content?</th>
<th>Mass Market or Elite?</th>
<th>Users can compete head-to-head</th>
<th>Integrates with VCR-DVD &amp; CD-MP3</th>
<th>Integrates with PC or consoles?</th>
<th># of Brands Supporting Interface</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>No</td>
<td>Elite</td>
<td>No</td>
<td>No</td>
<td>No</td>
<td>1</td>
</tr>
<tr>
<td>B</td>
<td>No</td>
<td>Elite</td>
<td>No</td>
<td>No</td>
<td>No</td>
<td>1</td>
</tr>
<tr>
<td>HeartRate</td>
<td>Yes</td>
<td>Both</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Both</td>
</tr>
<tr>
<td>D</td>
<td>No</td>
<td>Both</td>
<td>No</td>
<td>Yes</td>
<td>No</td>
<td>4</td>
</tr>
<tr>
<td>E</td>
<td>Some</td>
<td>Elite</td>
<td>Yes</td>
<td>No</td>
<td>PC-only</td>
<td>1</td>
</tr>
<tr>
<td>F</td>
<td>No</td>
<td>Elite</td>
<td>Yes</td>
<td>No</td>
<td>PC-only</td>
<td>1</td>
</tr>
</tbody>
</table>

*Fig. 18*
### Fig. 19

<table>
<thead>
<tr>
<th>Feature</th>
<th>iExercise</th>
<th>Others</th>
</tr>
</thead>
<tbody>
<tr>
<td>Controls exercise equipment</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>Works with workout music &amp; videos</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>Allows automatic upload of exercise data</td>
<td>Yes</td>
<td>No</td>
</tr>
<tr>
<td>Enables true interactive games</td>
<td>Yes</td>
<td>No</td>
</tr>
<tr>
<td>Allows users to compete head-to-head</td>
<td>Yes</td>
<td>No</td>
</tr>
<tr>
<td>Connects to PCs &amp; game consoles</td>
<td>Yes</td>
<td>No</td>
</tr>
<tr>
<td>Available to all brands of equipment</td>
<td>Yes</td>
<td>No</td>
</tr>
</tbody>
</table>

### Fig. 20

[Diagram showing interconnection hardware, iExercise hardware interface, USB chipset & firmware, game console (Xbox, PS3, or PC), iExercise USB driver, iExercise APIs, exercise application, and AV connector to television.]
Fig. 23
Fig. 24
Fig. 25
USER INTERFACE AND METHODS OF USING IN EXERCISE EQUIPMENT

PRIORITY CLAIM

[0001] This application claims priority to and incorporates by reference in its entirety U.S. Patent Provisional Application No. 60/895,676 filed Mar. 19, 2007. This application is a continuation-in-part of U.S. patent application Ser. No. 11/078,913 filed Mar. 9, 2005 that in turn claims priority to U.S. Patent Provisional Application No. 60/551,366 filed Mar. 9, 2004. All above applications are incorporated by reference in their entirety.

FIELD OF THE INVENTION

[0002] The invention relates generally to graphically based interfaces.

BACKGROUND OF THE INVENTION

[0003] The average user's usage of exercise equipment, especially cardio workout equipment involving running or similar activities lessens with time. Exercise equipment is designed by sports physiologists with a penchant for displaying alphanumeric physiological parameters in view of the exercising user. Some of the displays are augmented by peak activity bar charts and caloric burn rates. After a short while, the user's interest in these alphanumeric and chart plots lessens and interest in exercising soon tapers off.

[0004] The decline in exercise equipment use is related to the boredom that sets in from the inherent repetitious aspect of treadmills, stationary bikes, elliptical trainers, and other cardiac related equipment and the commonness presented by the data and plot displays. Exercise equipment users try their best in alleviating boredom, for example listening to portable music devices or watch a nearby wall mounted television with limited success.

[0005] Currently people are forced to exercise indoors on fitness equipment without any interactivity, resulting in a boring experience that requires significant willpower to start and continue, resulting in considerable lack of motivation to keep exercising in shape, much less form healthy exercise habits. In the particular embodiment, the problem is solved by turning exercising on a piece of fitness equipment into an interactive, exciting, and motivating video game experience.

[0006] Another existing problem is active or fitness games have yet to penetrate the home in any significant degree. While an arcade game can be successful enough to support a proprietary solution, no single game, or even small set of games, is sufficient to motivate the consumer to make a substantial hardware purchase. Industry-wide support must be shown before most consumers can adopt a platform.

[0007] The current market solutions are inadequate. Proprietary solutions generally target niche markets and may consist of a proprietary hardware device, such as a bicycle or treadmill, an interface, and some proprietary software that runs the application; the user is required to supply the platform, usually a personal computer (PC). These solutions are closed, meaning there is absolutely no support for their devices other than from a few choice partners, and their costs force pricing high far above the mass-market.

[0008] Other approaches rely on the use of existing games that do not work off-the-shelf as fitness games. Further, because the devices do not provide a way to equally compare the effort put forth by two competitors, these devices cannot be used for head-to-head competition or even friendly training.

[0009] In prior devices, there is no feedback from the device, so they cannot be used for interactive games or competitive events, and the settings of speed and incline or resistance are absolute, not relative, which means that a given workout is only good for someone within a narrow fitness range. Someone more fit than that range will find the workout too easy, for someone less fit the workout is too hard.

[0010] In societies afflicted with obesity, diabetes, coronary heart disease, high blood pressure, and other diseases or conditions associated with or exacerbated by a sedentary lifestyle, overcoming the boredom effect inherent in cardio workout equipment so that usage levels and frequency is restored to reach and surpass health maintenance levels reflects a goal worthy to achieve.

SUMMARY OF THE INVENTION

[0011] An application programmable interface (API) configured to present a selection of video games having interactive gaming content responsive to a variety of physical motion of the exercise equipment that is powered by the equipment user and designed to encourage ongoing participation by the user in various physical activity scenarios. The API is configured to merge exercise equipment with video gaming that is interoperatively compatible across multiple exerciser hardware configurations, across multiple gaming system equipment configurations, and across independent display monitors. Gaming content is designed to interactively encourage continued and varying exercising by associating the motions of the exercise equipment with changes in audio visual content presented on the monitor or by other gaming system configurations.

[0012] The API serves to configure fitness equipment to be generically interactive and interfaceable with arcade like games. The generic interactivity includes compatibility with multiple motion types generated by a user. For example, a rear-wheel trainer is coupled with a bicycle and interfaced to a personal computer (PC). The fitness equipment interaction happens in concert between the fitness hardware and a host device, e.g., a PC, game console, or other hardware and/or software device or system capable of implementing an interactive application. Alternative embodiments include the API configuring other physical exercise equipment with an arcade game like experience, including a stair machine, a treadmill, an elliptical trainer, a rotary climbing wall, grasping activities, pulling, pushing, and other specially designed fitness equipment.

BRIEF DESCRIPTION OF THE DRAWINGS

[0013] Particular and alternative embodiments of the present invention are described in detail below with reference to the following drawings:

[0014] FIG. 1 illustrates a conventional bicycle installed with a rear-wheel trainer;

[0015] FIG. 2 illustrates the brow cover of the fly wheel on a rear-wheel trainer;

[0016] FIG. 3 illustrates a view of the fly wheel with the cover removed;

[0017] FIG. 4 illustrates a view of the inside of the fly wheel cover showing the reflective infrared sensor;
FIG. 5 illustrates a close-up view of the assembly under the front wheel;

FIG. 6 depicts the start of an example interactive fitness game;

FIG. 7 depicts an example racing game;

FIG. 8 shows the car accelerating through a turn of the racing game;

FIG. 9 illustrates an overall layout of the particular software solution, allowing fitness manufacturers, game developers, and fitness professionals to understand the solution and how to work with it;

FIG. 10 illustrates how a particular embodiment allows any device to work with any application, without requiring any modification or work by the end user; this makes a particular embodiment of applications and devices “Plug and Play”, preferably by using a runtime core that removes all dependencies between devices and applications;

FIG. 11 illustrates an alternative embodiment of the typical functions used and the typical path followed through the system;

FIG. 12 illustrates a particular embodiment runtime core that preferably removes all dependencies between devices and applications involving a music control;

FIG. 13 illustrates an internal design which optionally allows multiple device types to be integrated and made equivalent in an equivalence layer, while simultaneously preserving device specific information in exercise records and elsewhere to meet the user’s needs and for future expansion;

FIG. 14 illustrates a particular embodiment exemplarily for interactive exercise devices;

FIG. 15 illustrates a fitness related equipment attached with a game console or PC with an enabled application, which then may be connected via an AV cable to a television or other audio video device;

FIG. 16 illustrates a another particular embodiment’s technical baseline;

FIG. 17 illustrates how a particular embodiment includes fitness, entertainment, and provides a user with many features and opportunities;

FIG. 18 illustrates comparative products and features;

FIG. 19 illustrates advantages of another particular embodiment;

FIG. 20 illustrates an alternate API platform and interconnectivity;

FIG. 21 schematically depicts features of non-HRG interfaces;

FIG. 22 schematically depicts the more numerous features offered by HRG interfaces;

FIG. 23 schematically depicts the services available from HRG research and development in providing an integrated exercise and entertainment experience;

FIG. 24 schematically depicts the integration of marketing services with other HRG services; and

FIG. 25 schematically depicts the communication protocol options between HRG Internet Services and fitness machines interfaces with entertainment consoles via HRG’s application programming interface.

DETAILED DESCRIPTION OF THE PARTICULAR EMBODIMENTS

The present invention relates generally to an interface adaptable to exercise equipment systems that helps encourage exercise equipment use to levels that is consistent with reducing obesity and to mitigate the effects of other chronic illnesses.

Description below concern an application programmable interface (API) configured to present a selection of video games having interactive gaming content responsive to the physical motion of the exercise equipment that is powered by the equipment user. In one embodiment, the interactive video games are displayed on a monitor through the API that is interoperatively compatible with multiple exercise equipment platforms. In another embodiment, the API is interoperatively configured to present a range of gaming scenarios through video game systems in communication with their respective display monitors in which the gaming content therein is interactively responsive via the API with the multiple exercise equipment platforms. The video API and its accompanying graphic user interface (GUI) provides an interactive entertainment experience to the exercise equipment user, thereby motivating the user to keep actively exercising with the exercise equipment to achieve a desired fitness level using arcade-like games.

Descriptions below also further delineate the merging of exercise equipment with video gaming through the API that is interoperatively compatible across multiple exerciser hardware configurations, across multiple gaming system equipment configurations, and across independent displays. Other embodiments provide for interoperatively communicating between exercise equipment and display monitors to provide gaming content designed to interactively encourage continued exercising by associating the motions of the exercise equipment with changes in audio visual content presented on the monitor.

Particular embodiments of the disclosure provide for a computer readable medium having executable instructions for a method to play an interactive game presentable on a display and associated with an exercise device operated by a user. The computer executable instructions include a method of installing an application programmable interface (API) that is capable of presenting gaming content of the interactive game that are modifiable by the conveyed motion that is generated by the user operating the exercise device. The API may be configured for receiving signals proportional to the conveyed motion, associating the signals to at least one symbolic representation of the interactive game, processing the signals in proportion to parameters defined by the gaming content, and changing the status of the symbolic representation within the interactive game in proportion to the signal processing.

The computer readable medium includes instructions for making the API compatible to receive signals proportional to the conveyed motion includes motion that is associated with translation, rotation, sliding, stepping, grasping, lifting, squeezing, pulling, stretching, pushing, and jumping. The instructions also provide for associating the signals to at least one symbolic representation includes a mechanical depiction, a life form depiction, and a graphic icon. The symbolic representations may occur in gaming contents of a self-competing or interpersonal competing nature.

Alternate embodiments provide that the API includes at least one of a physics interface, an intensity interface, a device equivalence interface, a treadmill device interface, a bicycle device interface, and a generic device interface interposed between an application program presenting the
gaming content and the exercise device. The device equivalence interface allows diverse gaming applications to communicate between classes of devices, for example, treadmills and bicycles, or other fitness machines having different designs. The device equivalence interface is adaptable creating multi-player games, either locally launched or downloaded from online fitness training entities. The device equivalence interface allows groups of multiple gaming users and/or their coaches to participate in online exercise-gaming competitions. The online exercise-gaming competitions include bicycling, running, half-side spinning, skateboarding, climbing, and simulated sporting and combat weapons movement. Through the active involvement of gaming users and/or their coaches, a large unified audience of fitness-gaming entertainment enthusiasts is generated and expanded by through utilization of the API’s device equivalence interface. The growing audience of fitness-gaming enthusiasts provides demand for the creation and downloading of other entertainment games that can be adapted to the exercise machines via the device equivalence interface.

[0045] Other alternate embodiments provide that the mechanical depictions may include automobiles, ships, airplanes, spacecraft, submarines, and sports equipment, the life form depictions to include animal, vegetable, human, microbial, and imaginary, and the graphic icon to include arcade game icons. The sporting equipment may include baseball, football, hunting, and fencing. Yet, other embodiments may provide for the changing the status of the symbolic representation within the interactive game to include changing at least one of the appearance and location of the mechanical depiction, the life form depiction, and the graphic icon within the gaming content of the interactive game. The API may also be capable of reading a the computer readable medium having code for an interactive exercise program, and be configurable to present a video overlay, the video overlay containing at least one of a current heart rate, current exercise intensity, upcoming changes, time and distance.

[0046] Other embodiments provide for a method for using a user interactive exercise system that includes operating an exercise apparatus, capturing motion-related data derived from the exercise apparatus, transmitting the motion-related data to a device capable of displaying a gaming scenario having a symbolic representation, and manipulating the graphic representation in proportion to the captured motion-related data. The motion-related data may include data attributable to translation, rotation, sliding, stepping, grasping, lifting, squeezing, pulling, stretching, pushing, and jumping.

[0047] In one particular embodiment, the API integrates cardio fitness equipment with home entertainment systems. The home entertainment systems may include game consoles from competing manufactures, allowing users to integrate exercise into home entertainment activities. The cardio fitness equipment includes treadmills, elliptical trainers, stair steppers, and stationary bikes. The exercise effort and/or other physiological inputs can be inputted through the cardio fitness equipment to derive a gaming experience. In a stationary bike embodiment, the exercisers pedal and/or steer their way through gaming scenarios, winning bonus points and/or awards for achieving exercise targets. Other embodiments of the stationary bike exerciser provides for game content downloading from a remote or local server, game parameter uploading to the remote or local server, team play and competition between remote or nearby exercise users under real or present time conditions or time-delayed conditions against a player who has already uploaded gaming data to the remote or local server accessible by currently engaged exercisers.

[0048] In other embodiments gaming data of competing participants may be uploaded to nearby or remote servers for health data storage and analysis. The data then may be retrieved, and based on analytical programs, gaming participants may receive prescriptions of various online health and fitness programs, receive delivery of customized health and fitness content within newly delivered entertainment software downloads, and obtain care provider support or opinions from user communities.

[0049] The online connectivity within the user communities provides users with real-time or present time play and/or competition, the ability to purchase and enjoy a continuing stream of fun, new games or entertainment environments, and services related to tracking or personal-training podcasts.

[0050] The application programming interface (API) includes a fitness equipment hardware interface, gaming programming configured to be operable by the API, and a plurality of games ranging from casual, low activity to fully engaged maximal activity expanded by a user with the fitness equipment. The gaming content may include movement enabled in which the storyline and/or difficulty levels may dynamically change by the intricacies and speed exhibited by the user’s movement conveyed by the user operating the fitness equipment hardware and through the API.

[0051] A particular embodiment comprises one or more of the following: a fitness device (aka fitness machine, e.g., treadmill, stationary bicycle); a user display; a computer system for running a game (simulation or other software, e.g., PC, game console, or handheld device). Alternate embodiments may also include a method of detecting the level of effort of the user (e.g., paperweight watts of output); and/or an optional method for monitoring the user (e.g., their perceived level of exertion).

[0052] The particular embodiment can provide an interface that allows any fitness machine to look like a single-axis joystick. In an alternative embodiment, multiple additional axes are possible (e.g., steering on a bicycle). The particular embodiment may include buttons and sliders that may be reported out to common USB Human Interface Device (HID) spec drivers. The particular embodiment may have connectivity with a PC, Xbox, PS2, Game Cube, etc. In an alternative embodiment, two-way control (e.g., force feedback, increased device speed or resistance) can be available as well.

[0053] The particular embodiment includes an interface that allows control of any fitness machine from an Application Programmer’s Interface (API) which may run on top of a manufacturer’s published or non-published interfaces. Examples of such interfaces: Icon’s fit interface (e.g., chipmaker code); and Precor et al.’s CSafe interface (e.g., HRG code); direct control of a machine using the equivalent of the LabJack USB controls for our Torcs demo (e.g., HRG code, LabJack USB driver code and hardware). All code mentioned in this paragraph is herein incorporated by reference.

[0054] In the particular embodiment, the code resides on the device itself, in an interface pod, or on any other part or combination of parts of the fitness solution, including any main CPU.

[0055] The particular embodiment optionally includes a calibration of level of effort between machines: of a single type, e.g., two machines of the same type from the same manufacturer, of same type but different manufacturers, e.g., two different stationary bicycles; and of different type and
classes, e.g., a stationary bicycle and a treadmill. Equivalence between users on the same machine, or on machines made
equivalent through the use of this interface allows a “fair fight” no matter what the two machines are, thus allowing for
multi-person sessions locally or over a network.

[0056] The particular embodiment may further allow automatic adjustment to machine settings, including level of
effort. First, for an experienced user on a new machine e.g.,
doesn't have to know how to setup a machine he or she hasn't
used before because the system knows specific characteristics
of the user and can compute the appropriate settings. Second,
between a user and a user's exercise plan, the interface knows
sufficient data about the exercise plan and the level of effort.
Third, for a naive user, who needs their fitness level evaluated
the system can automatically adjust to an appropriate baseline
based on data from one or more sessions. Finally for a user
against AI or other elements of a game or other simulation
system, adjusting the user's capabilities to a known baseline
appropriate for a designer's desired level of difficulty. Appen-
dix A provides a code list for implementing the aforementioned
particular embodiments.

[0057] In another particular embodiment, the traditional AI
balance technique of “Rubber banding” is expanded to pro-
vide more features. Unlike in a traditional game, where
"throttle is free", i.e., it costs the user no effort to hold the
throttle wide open for long periods of time, for an exercise
game or simulation to be "fair" and "beatable" the system
needs to know the actual potential of the user and the system.
For example, a typical car racing game would speed up the AI
cars the user is racing against when the user is going fast,
allowing them to catch up, and slow them down somewhat
when the user is behind, mounting the cars on a "rubber band"
relative to the user's car. Since there is no "cost" to the user
to go as fast as possible this is a reasonable technique in a
standard game. However, if an exercise system were to imple-
ment such a simple system, the users would quickly learn that
they should go as slow as possible for the majority of the race,
speeding up only near the end. This would result in an exer-
cise system that was neither a fun nor effective. Instead, the
system is designed to be able to change the dynamics of the
"fitness solution" relative to the characteristics of each indi-
vidual exerciser. In order to accomplish this task the system
acquires knowledge of the user, and/or a good muscle/skeletal
model of human output that understands quickly the individ-
ual's ability. The particular embodiment executes an exam-
ple of this process by using code in the _speed_limiter.cpp which
works by modeling physical dynamics of an actual machine,
e.g., power/RPM curve of a bicycle trainer, using a straight
watts equivalent, to control the maximum speed of the AI cars
along with their aggressive or passive nature (e.g., for pass-
ing), based on how well or poorly the user is doing. Alterna-
tive embodiments may use programming models including:
Bicycle (power/speed) or Treadmill/Walking (speed/weight).
Appendix B provides a code listing to implement a
particular speed limiter feature.

[0058] In other embodiments there is an additional level of
input massaging: braking (an additional joystick axis) gener-
atored automatically by processing the "input signal" from
the rotation of the rear wheel or equivalent energy absorption/
storage/dissipate device as demonstrated in the demo with
bicycle on standard rear-wheel trainer. Braking is preferably
included because in order for a more realistic and engaging
experience the user should be able to slow down as well as
speed up.

[0059] For example, input from human is modeled from
knowledge of the mechanical nature of the human body in one
or more of the following ways: baseline (DC) offset plus A/C
signal (treating both legs as one input), baseline (DC) offset
plus two A/C signals (treating each leg separately), two A/C
signals (treating legs either together or separately).

[0060] Alternatively, additional models are derivable for all
types of fitness machines (e.g., rowing, elliptical trainers,
treadmills, etc.), based on knowing or assuming the power
characteristics of the dissipative device (typically exponential
with respect to the current short-term average input power for
the human).

[0061] The particular embodiment can also detect changes
in output energy (e.g., constant, increases, or decays, or
decays rapidly) and thereby can distinguish or separate:
acceleration by user, braking with hand brake on rear wheel,
and coasting steady-state pedaling/running/rowing/etc.
Appendix C lists a code implementing the brake detection
feature.

[0062] Alternative embodiments of inputs include total
weight of user, and weight shifting left/right, front/back, or
up/down unweighting and weighting which provide a
increasingly realistic and meaningful experience to the user.
An alternative embodiment may include detection of non-
human input assists e.g., cheater detection.

[0063] The particular embodiment includes a system that
tracks the user and provides multiple levels of support: cross
platform data reporting and analysis, understands exercise
plans and provides guidance and control of the “fitness solu-
tion” about the user's needs for the current exercise session
(e.g., during the exercise session, notifies the "fitness solution"
about: warm-up, cool down, and interval starts & ends. The
particular embodiment allows user to modify and store modifi-
cations to exercise plan either for this exercise session or the
long-term plan, or both.

[0064] Another particular embodiment allows the system
to learn, through integrated or off-line analysis, the user's
level of fitness which may be mediated/controlled by a coach,
trainer, nurse, doctor or other professional. The particular
embodiment can provide automatic changes to exercise
plans.

[0065] The particular embodiment provides for finding
“equals” or appropriate competitive/cooperative partners for
multi-person fitness sessions that can be used to correlate a
person's reported level of fitness through the interface to
actual real-world fitness tests.

[0066] The particular embodiment allows understanding of
relative performance at in-person competitive events as com-
pared to on-line and/or solo events. Preferably, the system
can validate “real” users vs. cheaters.

[0067] An alternative embodiment allows extensions to the
system, e.g., a RFID-encoded heart rate monitor, or the pres-
ence of an external device, such as card-key or cell phone, that
allow users to be automatically identified to a fitness solution
along with any other particular or optional peripheral, such as
body state monitors (e.g., heart rate, blood pressure, blood
oxygen level, ECG/EKG, weight, body fat content, oxygen
uptake, etc.)

[0068] The particular embodiment can include sensors,
actuators, etc., and their relevant interfaces which use any
communication mechanism to the host, e.g., a wireless inter-
face such as Bluetooth™, 802.11b, wireless USB, etc; a wired
interface such as USB or Firewire™, etc; a fiber-optic inter-
face; or any other interface capable of enabling communication between a sensor and the host.

[0069] The particular embodiment consists of a software Application Programming Interface (API) and a USB-compliant hardware interface to fitness devices. This allows developers to create games for the PC, PS2, XBox, and other platforms and to use fitness equipment compliant with the particular embodiment as easily as they would any other input device.

[0070] The particular embodiment is designed for anybody who wants to be in better shape. Equipment enabled with the particular embodiment changes the way people exercise: fun exercise that replaces boredom; and the ability to track and manage historical workout activity. For fitness equipment vendors, the particular embodiment provides: product differentiation among other equipment OEMs and against existing products and drives replacement of existing equipment by existing customers. For workout and game publishers, the particular embodiment enables a new interactive game platform: new life for existing workout video and music lines; and new life for existing games appropriate for exercise modalities. It provides a new game genre and market opportunity, interactive fitness games.

[0071] Because the particular embodiment is provided by an independent company, preferably with expertise in middleware software, rather than a fitness equipment manufacturer or a proprietary game platform, it is far more likely to be received warmly rather than as a competitive threat.

[0072] The particular embodiment is a benefit for consumers because the particular embodiment represents a trusted interface brand, à la VHS tapes—they know that any iExercise-labeled game can work with any iExercise-labeled fitness device, so they can shop for both games and devices without worrying about technical details. USB interfaces are provided on the Xbox, PlayStation2, and the PC, so consumers do not need to buy a new computer or game console. iExercise is a currently contemplated trademark for certain versions of the particular embodiment. However, the scope of the particular embodiments are not to be limited by this trademark or any other.

[0073] In the particular embodiment the cost of adding USB interface hardware to the exercise device is trivial, so price, configuration, and availability are not obstacles for iExercise as they are for proprietary solutions. Consumers simply plug the USB cable from the device into their console or PC, insert the game disk and they are up and running.

[0074] The particular embodiment has a broad game catalog that relates to market success of iExercise. Additionally, a selection of iExercise devices wide enough to appeal to most consumers would be available. The particular embodiment reduces cost, especially of hardware implementation, maximizes market potential, minimizes time to market for products, minimizes development requirements for iExercise products, supports legacy future hardware, and has no platform dependencies on the device side (i.e., all devices preferably work on all platforms, and/or maximize developer flexibility).

[0075] In the particular embodiment, both the elite and mass market are particular customer bases. Elite users already consume devices, but the mass market can be the goal of most developers. Therefore, preferably, while the most critical features of the elite user should be implemented, the baseline criteria are ease of use, i.e., removal of obstacles to adoption and continued use.

[0076] Preferably, the system is easy to use—for any game, with any device, the user can insert the game, plug in the device, and immediately begin playing without having to do any advanced configuration, e.g., log in, set up an exercise program, configure the device, etc. Preferably, reasonable defaults are provided for all configuration parameters.

[0077] In an alternative embodiment, the following features can be added for advanced users: a calendar with exercise scheduling and reminders, results reporting and tracking, and training advice and help.

[0078] In the particular embodiment several technical features have been distilled from the marketing requirements: Simplify development of fitness games, keep the developer from having to know about exercise physiology, applications and equipment may work on “first plug-in” without user intervention, ease of use is paramount. All of the above implies that fitness equipment should be more or less completely hidden from the user. This abstraction includes: device detail hiding, device equivalence e.g. any device can be used in place of any other, and user hiding e.g. while it is acceptable for the application to know about the user, the application may be able to run without any knowledge of the user.

[0079] In the particular embodiment, fitness equipment manufacturers are afforded as much flexibility in producing new hardware as game developers do software. This means that, in the particular embodiment any device can interface to iExercise USB hardware without driver changes, upgrades to devices may not require driver changes, and hardware interface requirements may not cause significant cost increases.

[0080] In the particular embodiment users optionally have the following functionality available: scheduling exercise, reporting exercise results, tracking exercise results, enter the user information, e.g., weight and heart rate information, and allow for fitness testing.

[0081] In the particular embodiment the API is built in several layers in order to achieve the following: offer maximum flexibility to developers by allowing them to choose the appropriate interface(s) for their application, create a dynamic system of extensible components that can accommodate future devices and application needs, allow modules to be designed, tested and updated independently.

[0082] In the particular embodiment layers allow the interface to support all fitness equipment types, yet provide a simple programming model for the application. This allows rapid development of applications that automatically support all devices.

[0083] In the particular embodiment it is anticipated that iExercise can appeal to a wide variety of developers with different skills, resources, and visions. There are the applications such as bike racing, virtual tours of cities, marathons training; a simple application on a cell phone or PDA to control the treadmill at the gym and record your workout, or a game of bike-powered Tetris are examples of more obscure uses.

[0084] The particular embodiment employs a physics model that may be appropriate for any application that models “the real” world. It allows the application to set incline (slope), desired speed, and various types of resistance such as wind resistance and rolling (tire) resistance.

[0085] Another particular embodiment employs an intensity model that may be appropriate for any application that just wants to make the player work harder at certain times than others. For instance, in a “Tetris” style falling blocks game,
the game can increase the intensity near the end of a round. If the player walks, runs, or pedals faster than the intensity, then the rate of full is slowed; otherwise, it is increased.

Alternate embodiments employ a combination of “intensity” and “physics” and/or other models. In alternative embodiments, as new genres of applications arise and specific needs are identified, new interfaces can easily be added alongside the physics and intensity interfaces.

In the particular embodiment, a device equivalence interface model is designed to allow an application to respond in a similar fashion across devices such as treadmills and bicycles. The mapping functionality provided in this interface can convert parameters from the various devices into a common unit such as power, thus providing an “equivalent” experience.

One embodiment includes a simple interface that provides functionality specific to a treadmill. Another embodiment includes a simple interface that provides functionality specific to bicycle.

In the particular embodiment the generic device interface is the lowest layer in the architecture that may be useful for developers. As the foundation of the system, this interface was written to support all fitness devices that can be represented by a set of scalar and vector values. This interface provides a general solution to detect hardware, query it for available parameters, their access permissions, and then allows the user to set and retrieve these values. The items of interest when using a treadmill, for example, would be speed, incline and time elapsed; with a bicycle one might want speed and a resistance profile. This interface also provides functionality for device synchronization, and the registration of callbacks used for notification purposes. The level of abstraction in this layer shields the user from the various communication protocols employed by different fitness devices and presents a uniform interface that accommodates future devices.

In alternative embodiments iExercise provides additional interfaces that include the following functionalities: Fitness device detection and enumeration, simulation of terrain segments expressed in terms of distance and gradient, recording of session parameters such as heart rate, distance traveled, calories expended, workout duration etc., and miscellaneous clocks and timers. In alternative embodiments, several common supporting interfaces are available. These allow access to advanced functionality and information about the user.

In the particular embodiments, the interval callback allows the application to support user-specified exercise sessions. The user can set features such as minimum warm-up time, number of intervals, interval length, time or recovery requirements (e.g., heart rate below some rate) between intervals, and minimum warm-down time. Applications registering for interval callbacks can be provided with this data, as well as callbacks when intervals should begin and end. In this embodiment the exercise data, such as miles traveled, calories burned, average intensity, etc. are all available through the data interface. In the particular embodiment override, callbacks are provided for several cases. These cases include the user’s changing a device setting, such as incline or speed, or some other condition as indicated in the result code.

In the particular embodiment, the application may not be asked to access user data. However, it is available to the application if desired. The user data structure is as defined in the interface header files.

In the particular embodiment a data-acquisition module (DAM) forms the interface between the exercise device and the iExercise middleware. Sensors on the exercise device monitor variables such as speed, heart rate, incline, etc. and these are fed into the DAM that in turn provides these to the iExercise API. The DAM is compatible with fitness equipment from the major manufacturers and provides a seamless interface across diverse hardware platforms.

The iExercise API is preferably comprised of several closely related objects that can be used in different configurations to achieve the following: allow a USER to interact with a MACHINE within a given CONTEXT. The USER is a representation of a person, and to represent that person, information of the following kind is needed: Height, weight, age, sex, fitness level, workout goals etc. The MACHINE is a representation of a device such as a bike, a treadmill, an elliptical trainer, etc. and to represent that device, information including speed, incline, resistance, heart rate, power, etc may be needed. The kind of information differs by device according to the motion types and duration. The CONTEXT is a representation of a particular application that the USER is immersed in, and could be a race simulation, a shooting game, and/or a weight-loss workout. The iExercise API currently defines the following functionality. Appendix D lists the code for implementing the iExercise API.

Other embodiments described below depict a selection of video games having interactive gaming content responsive to the physical motion of the exercise equipment that is powered by the equipment user. In one embodiment, the interactive video games are displayed on a monitor through the API that is interoperatively compatible with multiple exercise equipment platforms. In another embodiment, the API is interoperatively configured to present a range of gaming scenarios through video game systems in communication with their respective display monitors in which the gaming content therein is interactively responsive via the API with the multiple exercise equipment platforms. The video API and its accompanying graphic user interface (GUI) provide an interactive entertainment experience to the exercise equipment user, thereby motivating the user to keep actively exercising with the exercise equipment to achieve a desired fitness level.

The particular embodiment enriches the fitness/entertainment experience by taking the human factor into account. The users of exercise equipment come in all shapes and sizes, have different skill levels and levels of fitness, and have differing needs. The particular embodiment provides application developers with much information about the user as is needed in order to create content that is engaging and in which game-play is fair when matching opponents with different skill levels.

The particular embodiment can combine USB-compliant hardware and a high-level software API that provides a cross-platform solution for creating games and other applications for fitness equipment such as bicycles and treadmills. This solution that can be compatible with PCs, game consoles, and a wide variety of fitness devices, offers application developers the opportunity to adopt a non-proprietary, cross-platform system.

The particular embodiment changes the way people exercise: Fun exercise that replaces boredom; and the ability to track and manage historical workout activity. For fitness equipment vendors, the particular embodiment provides: Product differentiation among other equipment OEMs and
against existing products drives replacement of existing equipment by current customers. For workout and game publishers, the particular embodiment enables a new interactive game platform: New life for existing games appropriate for exercise modalities, A new game genre and market opportunity—interactive fitness games.

[0099] The particular embodiment makes the cost of doing an exercise modality trivial relative to content development costs, and the cost of integrating devices into fitness hardware also trivial.

[0100] In alternative embodiments many different modalities are possible for many different games all have common display of current intensity, intensity, heart rate, optional display of upcoming intensity, time elapsed, time remaining, etc. In some embodiments modalities for race simulations are provided.

[0101] In an alternative embodiment there is a first-person shooter modality where you have to pedal above a minimum threshold to get more ammo, the faster you pedal, the faster you can move, and at max pedaling, can get special weapons. In an alternative embodiment there is a “Tetris”-style modality which can flow falling block by pedaling faster, “intervals” possible as blocks fall faster, “reserve” energy allows you to destroy bottom layers. In an alternative embodiment there is a “Myst”-style modality where you have to pedal to travel around the place being explored and certain devices may be “charged.”

[0102] In the particular embodiment exercise profile initialization supports the combination of several methods of setting up initial profiles for a given user: standard defaults, either fully generic or based on upon inputs such as age, sex, and current exercise habits, medical limitations, coaching or therapeutic guidance, guidance for fitness, weight loss, etc., and other guidance appropriate to exercise plans.

[0103] Preferably, the runtime core executes most of the real-time processing, including: device equivalence; monitoring the user as indicated by the fitness plan; communication with the game, including: intensity intervals, warm-up & cool-down, current user intensity & power, and information about the user & fitness solution for display or other purposes, e.g., name, device type, etc.; and, communication with the device driver: any setup for this user and this device, device capabilities, including information to compute instantaneous watts being generated by the user, and information about the device that may be desired by the fitness records, the game, or other needs.

[0104] The particular Universal Device Driver is a relatively large body of software compared to many other device drivers. It is charged with understanding how to communicate with the various exercise devices. Preferably, it does this through several paths: it knows how to communicate with devices supporting existing published and unpublished standards, such as CSafe and iFit, it supports a generic set of interfaces, known collectively as the iExercise Device Compliance specification, which provide standard methods for reading device state (e.g., instantaneous power output of the user), device capabilities (e.g., power curves and meaning of resistance settings), and device-specific information (e.g., a string specifying the name of the device); and standard methods for writing to the device (e.g., for setting resistance settings or other standard user setup.)

[0105] In the particular embodiment an iExercise compliant game is simply any application, be it a true game, an exercise or race simulation, or other application that uses the iExercise API to provide a meaningful experience to a user (or users, in the case of multiplayer applications) of a fitness solution.

[0106] In the particular embodiment an iExercise compliant device is any device supporting: the iExercise compliance specification; or a standard interface that iExercise knows how to talk with, such as CSafe (Precor et al) or iFit (Icon et al); or a direct hardware connection (such as in the demonstration system) to a USB interface. All code and interfaces mentioned in this paragraph are herein incorporated by reference.

[0107] In the particular embodiment a “Happy User” is any person using a “Fitness Solution” described herein for an exercise session. The described “Fitness Solution” as a whole provides for interesting exercise sessions which meet a variety of fitness or rehabilitation goals and provide meaningful experiences to the user, said experiences providing many benefits, including some or all of the following for each user: excitement and engaging exercise rather than boredom, positive feedback about progress, reduce or eliminate completely the “willpower” to start and maintain an exercise program, and mental training (e.g., racing a specific Tour de France hill climb against Lance Armstrong’s best time ever) unavailable through traditional means.

[0108] The particular runtime core provides two particular elements for removing dependencies: (1) device equivalence conversion between the different device models supported by the Universal Device Driver and proper initialization of all device types based on direction of the training profile and activity software; and (2) API equivalence between applications, preferably by translating between multiple programming models attractive to various activity developers. Two examples are a pure intensity model, and a slope/weight/ speed model. The standard interface provides a standard training profile, thereby allowing exercise activities to be displaying without requiring a dedicated user interface for each session. Specialized interfaces may cause their own problems (e.g., user has to learn new interface for each activity, knowledge of previous exercise sessions would be lost or misinterpreted by each new activity, some exercise profile types may not be supported by all activities, activity developers aggress openers would have to be conversant with all the medical/athletic issues involved with exercise planning, and monitoring the user as indicated by the fitness plan).

[0109] In the particular embodiment, an iExercise compliant activity is a game, an exercise or race simulation, or other application that uses the iExercise API to provide a meaningful, realistic, engaging and immersive experience to a user (or users, in the case of multiplayer applications) of a fitness solution. Because these applications use the particular embodiment, they do not have to: build their own model of the user and the user’s needs, including all of the relevant UI and coaching/medical over-rides, support individually coded interfaces for every single device they want to support, and incorporate medical, athletic, and other training knowledge into the activity.

[0110] In the particular embodiment the iExercise Universal device driver handles communication between the Runtime core and the device itself. Thus, preferably, after device-specific initialization, which involves a handshake of the device-specific data, only a relatively small amount of data needs to be exchanged. This “run-time” data includes the setting of intensity level by the driver, and the reading from
the device data such as speed, additional axes of freedom, device supported user status such as heart rate, perceived exertion, oxygen uptake, etc.

[0111] FIG. 1 depicts a bicycle 20 installed in a rear-wheel trainer 21. Under the front wheel 22 is a device 23 that uses a joystick to measure how much the front wheel is or is not turned. The small red box 24 in front of that device is a USB interface 25 that reads the rear-wheel speed and transmits it to the host computer.

[0112] FIG. 2 depicts the blue cover of the fly wheel on a rear-wheel trainer 32. This is a commercially available rear-wheel trainer from Minoura that has been modified as a demonstration of the system. The wiring and small object at the bottom of the flywheel cover is for a reflective infrared sensor, which includes an infrared LED and pickup. The electronics for driving the LED are in the black box 31 hidden behind the leg of the rear-wheel trainer with the www.minoura.jp sticker.

[0113] FIG. 3 depicts the view of the flywheel with the cover 32 removed. The wiring connecting the infrared sensor is clearly visible, as is the black and white pattern 36 on the flywheel, which allows the sensor to detect the flywheel’s speed.

[0114] FIG. 4 depicts the view of the inside of the flywheel cover 32 showing the reflective infrared sensor 4. Preferably, this is a standard two part infrared sensor with an emitter & detector.

[0115] FIG. 5 illustrates a close-up view of the assembly under the front wheel 42, which is a device which uses a joystick 43 to measure how much the front wheel 42 is or is not turned. The small red box 41 in front of that device is a USB interface 44 which reads the rear-wheel speed and transmits it to the host computer.

[0116] FIG. 6 depicts an example of the start an interactive fitness game. The user on a bicycle is driving the yellow car 45 seen on the center of the screen 46. A two-axis “debug graph” 47 on the bottom right of the screen shows acceleration (positive vertical axis), braking (negative vertical axis) and turning (left and right horizontal axis). In this example, as the user pedals faster, the game model responds as if the car’s accelerator pedal was depressed in direct correlation to the speed of the rear wheel of the bicycle. If the user brakes with the bicycle’s brakes, the system detects this and applies the brakes of the car in proportion to the rate of slowing of the rear wheel, i.e., in proportion to how hard the user applies the brakes.

[0117] FIG. 7 depicts another example of the racing game. The user on a bicycle is driving the yellow car 45 seen in the center of the screen. The user has applied the brakes on the bicycle, which the system detects as shown by the small blue bar 50 in the negative vertical axis of the graph at the bottom right of the screen. Note the skid marks 51 just starting underneath the car.

[0118] FIG. 8 shows the car within the interactive fitness game accelerating through a turn. The user on the bicycle has turned the front wheel all the way to the left, as shown by the blue bar 55 in the horizontal axis of the graph at the bottom right of the screen, and is pedaling away as shown by the blue bar 55 in the positive vertical axis.

[0119] FIG. 9 illustrates a flowchart of the iExercise software solution. The iExercise software allows fitness manufacturers, game developers, and fitness professionals to understand the iExercise solution and how to work with it. The iExercise training profile program handles the updating of user training profiles based on the initial conditions set in Block 2, Exercise Profile Initialization, and the results of each exercise session as recorded by Block 3, the iExercise runtime core. Further explanation of the succeeding blocks is provided on the Figure itself (see 1-7).

[0120] FIG. 10 shows how the iExercise program illustrates a system flow chart for adapting exercise devices to work with fitness game applications. Adaptation is accomplished without requiring modifications or work by the end user, imparting to iExercise applications and devices a “Plug and Play” convenience. The iExercise training profile program handles the updating of user training profiles (initializing as described in FIG. 9) based on the results of each exercise session as recorded by Block 2, the iExercise runtime core. Data to do this is passed both ways between block 1 and block 2 as shown in the diagram. Further explanation of the succeeding blocks is provided on the Figure itself (see 1-4).

[0121] FIG. 11 is an alternative embodiment of the typical functions used and the typical path followed through the iExercise system. This figure further explains how a training profile is setup as it relates each compliant device and compliant application. Explanation of the flowchart nodes or blocks is provided on the Figure itself.

[0122] FIG. 12 shows the particular runtime core removes all dependencies between devices and applications involving an iExercise music control. This figure further illustrates how the particular embodiment allows the user to select music during all parts of the workout and allows the music to be faded in and out. Explanation of the flowchart nodes or blocks is provided on the Figure itself. The particular iExercise music control represents an extension class of the iExercise API as described in elements 2.1-2.3 of FIG. 13 (described below). The particular embodiment optionally includes this music control feature because music can be a valuable companion to exercise for many people, and this music control enables users to describe the “soundtracks” that should accompany their activity sessions.

[0123] FIG. 13 shows iExercise’s internal design which allows multiple device types to be integrated and made equivalent, preferably in a relatively simple equivalence layer, while simultaneously preserving device specific information in exercise records and elsewhere to meet the user’s needs and for future expansion. In the particular embodiment the iExercise training profile is defined in the class IX_CUserProfile and provides static methods for initializing the profile, getting profile information including data for the current session, and setting information such as the records generated during and after completion of the current session. In the particular embodiment class IX_CIntervalWorkout is one example of a class that is capable of reading a user profile and generating appropriate controls for an exercise session. This class generates signals to the application about when warm up, warm-down, intensity intervals should begin and end, time or other metrics for the end of recovery times, etc. 2.1 and 2.2 are left as placeholders for future classes for new workout supports (e.g., a fat burning workout) as well as new methods for interpreting historical training records (see further, box 7 and box 6, IX_CGradientCourse).

[0124] In the particular embodiment shown in FIG. 13, the IX_CSessionRecord captures the record of each session and is the method for examining each session record in all applications. IX_CHelperFunctions and IX_CGradientCourse can provide assistance to applications in decoding session records, as well as other functions that may be provided as extensions to the basic API.
[0125] In the particular embodiment shown in FIG. 13, the IX_CHelperFunctions provides a number of functions that most users of the API could write themselves, but are provided for convenience and correctness. Rather than placing these functions in random places in the API (where they might not appropriate, or might be hard for the user to find) a single place was provided for them.

[0126] In the particular embodiment shown in FIG. 13, the IX_CGradientCourse is the basic method of accessing the user’s output relative to a course described as a set of gradients, each with a slope and length. The gradient course may refer either to a historical course when looking at an exercise record, or to a current exercise session. A companion API, known as Energy API is also provided but not indicated on the diagram. The Energy API is analogous to the Gradient API but provides only momentary speed or energy records in its current version.

[0127] In the particular embodiment shown in FIG. 13, the IX_CThreadComm (for Communication) is responsible for communicating to treadmill class devices. As with the bike class, this class can be extended to include new devices that are comparable to treadmills. However, many devices, such as stair-steppers and elliptical trainers, may fit within the class without modification. However, entirely new classes may also be created (e.g., for isometric strength training machines.)

[0128] In the particular embodiment shown in FIG. 13, the IX_CDeviceAudio handles interaction with and communication to iFit-enabled bicycle devices. The IX_CBikeComm is the partner class to IX_CThreadComm, for stationary bicycle trainer class devices. Many other fitness devices, such as rowing machines, may end up in the bicycle trainer class. This classification is due to the resistance units common to all of them, which share the common attribute of having power-curves associated with determining watts of work being done by the exerciser. That is, an exerciser riding a bike at 10 miles an hour is doing far less than half of the work of somebody riding the same bike at 20 miles an hour (given the same resistance settings.) Likewise, somebody rowing at 2 miles an hour is doing less than half the work of somebody rowing twice as fast. In all cases the actual amount of work can be deduced from the specific speed/power curve of the device, which is to be specified in the device interface read at levels 15 and 16.

[0130] The IX_CDeviceEnumerator is responsible for finding all IX compatible devices available on the computer or console system that can run the exercise activity. It reports information about all such devices and is capable of returning specific devices that meet the user or activity specification.

[0131] The IX_CDeviceComm handles communication to all non-iFit devices. It can run on top of over types of device networks, such as USB, 1493 (FireWire), Bluetooth, and other device communication networks. It can also run on the device itself, if the device interface has a small microprocessor.

[0132] The IX_CDeviceAudio is the layer that handles communication to all iFit devices that rely on sound as the communication medium/protocol. In the particular embodiment shown in FIG. 13, the IX (iExercise) Hardware comprises the hardware for interfacing to and communicating with the device. As indicated in the description of box 13, this level can be moved with respect to the communication layer and the actual device. In the particular embodiment shown in FIG. 13, the IX (iExercise)-Compatible device is the device itself. All the device may need to be IX-Compatible is to support sufficient internal communication to respond to the device requests of box 13. The physical medium of communication (whether wired or wireless) is unimportant, as any transport layer can suffice for carrying the messages for iExercise. In the particular embodiment Multiple Devices/Multiple Users are handled at pre-initialization. If there are either multiple users or multiple devices, at initialization the activity is provided with data sufficient to allow the user to select who they are or what device they want. If the user is not found, default selection criteria are presented to allow a session without requiring the user to go through the user creation process. iExercise can save the training data in an "unassociated" slot with a user-defined tag for later recovery by the new user.

[0133] The various implementations of the particular embodiments provide for three fundamental experience "styles". In the first style, the game provides an environment that provides a reasonably consistent intensity. Mountain bike trail, running race course—the activity leads, setting intensity based on the environment within a range defined by the training profile and user history. Rowing—constant intensity set by user. iExercise suggests initial intensity based on training profile and user history. Activity tailors intensity to reflect the model of the physical dynamics of the activity. Game environment—iExercise leads, providing callbacks when sets change. Game is free to make intensity settings based on the game environment within a given range specified by iExercise. The iExercise activity indicates what style it, or the user, wants at startup. iExercise responds with the initial settings and the device parameters.

[0134] In another implementation the Common Interactions with Runtime Core is initialized in much the same way. The device descriptor is: Device Name: ASCII or Unicode character string; Device Type: iExercise-defined device type; and Device Capability: generic iExercise capabilities plus specific ones for device type. On Win32 platforms, the iExercise device driver can report as a DirectInput device and it is acceptable to deal with supplemental inputs (e.g., steering and brakes) through the DirectInput. This work has been done to allow devices with supplemental inputs to work directly as a joystick on all platforms, including Win32.

[0135] FIG. 14 shows exercise profile initialization, training profile program, runtime core, the UDD and, more generally how the particular embodiment sets the standard for interactive exercise devices. Explanation of the flowchart nodes or blocks is provided on the Figure itself.

[0136] FIG. 15 schematically depicts a particular embodiment in which fitness related equipment 93 and 94 attached, in the particular embodiment, to a game console or PC 92 with an iExercise enabled application, which then may be connected via an AV cable 91 to a television or other audio video device 90.

[0137] FIG. 16 schematically depicts an interface system chart having a plurality of equipment interfaces utilizing the
gaming application software 95 and an Exercise Compliant Device 104. The plurality of interfaces are adaptable to multiple exercise equipment types, and includes an Exercise Physics Interface 96, an Exercise Intensity Interface 97, an Exercise Device Equivalence Interface 98, an Exercise Treadmill Device Interface 100, an Exercise Bicycle Device Interface 101, and a Exercise Generic device interface 103. The Exercise Physics and Exercise Intensity Interfaces 96 and 97 communicate with the gaming application software 95. The Exercise Generic Device Interface 103 communicates with the Exercise compliant device 104. The Exercise Device Equivalence Interface 98 communicates indirectly with the gaming application 95 via either the Exercise Physics Interface 96 and/or the Exercise Intensity Interface 97. Exercise can present any device as any other device, so applications can always get the device type they want. This means that every device can be used with a plurality of experiences. The Exercise Treadmill Device Interface 100 and the Exercise Bicycle Device Interface 101 indirectly with the Exercise Compliant Device 104 via the Exercise Generic Device Interface 103. The Exercise Treadmill Device Interface 100 and the Exercise Bicycle Device Interface 101 also indirectly interface with the gaming application 95 via the Exercise Physics Interface 96 or the Exercise Intensity Interface 97, and/or via the Exercise Device Equivalence Interface 98.

The internal architecture of the HRG API, as shown in FIG. 16, illustrates the enablement of a single video game or other software application to be accessed in a uniform fashion by users of multiple device classes of fitness machines (e.g., treadmills and exercise bikes) or users of a similar device class of fitness machines that comprise different designs or are from different manufacturers.

The enclosed boxes 95 (Application) and 104 (Exercise Compliant Device) represent the API features that are exposed to either end-users, to application developers, or designers of Exercise compliant fitness devices. User exercise information is passed in a bidirectional manner between the Exercise Compliant Device and the Application through the internal components of the HRG API.

The dotted boxes represent the internal structure that enables the API to achieve the functionality stated above. This internal structure consists of the following elements: 103 (Exercise Generic Device Interface); Manages communication between Exercise Compliant Device and the rest of the API. Specifically directs communication between the Exercise Compliant Device and one of the device class-specific interfaces exemplified in box 100 and 101. Exercise Device Interfaces 100 (Exercise Treadmill Device Interface) and 101 (Exercise Bicycle Device Interface). These are device class-specific interfaces that optimize communication between applications and different device classes. The difference in fitness device classes is based on the type of exercise data used to calculate watts of output by an individual.

The Treadmill device class includes treadmills as well as stair climbers. This device class calculates exercise intensity using speed and incline information. In this class, watts of effort information may come from the fitness device, or via the HRG API, watts of effort can be calculated by the application given the user's weight combined with speed and incline information from the fitness device. The Bicycle device class includes stationary bicycles and rowing machines.

The Bicycle device class calculates exercise intensity using speed and resistance curve data, because stationary bicycle resistance is independent of the user's weight. In this class, watts of effort information may come from the fitness devices, or via the HRG API, watts of effort can be calculated by the application given the user's speed and resistance information from the bicycle, utilizing resistance curve information. This resistance curve information can be stored in read-only memory on the USB interface, within the application, or accessed online. Alternatively, the HRG API can calculate watts using speed information from the bike and a measurement of how quickly wheel speed decays at the top of each pedal stroke. In the HRG prototype, the resistance level is constant and the measurement of wheel speed decay is used to detect braking when the user uses the right brake lever to slow the racing car down while driving through turns on the racecourse. HRG support for other embodiments of exercise machines may be adapted to the system diagram depicted in FIG. 16.

Referring still to FIG. 16, the Exercise Physics Interface 96 and the Exercise Intensity Interface 97 provide initial links between the game or other software application and the HRG API. The Exercise Physics Interface 96 links the application to Exercise compliant fitness devices that are of the Treadmill-class, including treadmills and stair climbers. The Exercise Intensity Interface 97 links the application to Exercise compliant fitness devices that are of the Bicycle class, including bicycles and rowing machines. The interfaces 96 and 97 simplify the process of developing applications that are optimized for a specific device class. A developer creating a bicycle racing game can use the Exercise Intensity Interface 97 to optimize the game for Exercise-compliant exercise bicycles, knowing that the application won't have to calculate factors such as weight or incline that are involved in treadmill use. Alternatively, these 96 and 97 interfaces simplify the modification of existing games or version that are suitable for users of particular device classes. The video game Tetris, for example, contains no modeling of the physics environment experienced by treadmill users. Using the Exercise Physics Interface 96, a developer can more easily create a version of Tetris that incorporates treadmill physics concepts such as weight, incline, and watts.

As regards the Exercise Device Equivalence Interface 98, this interface enables application equivalence between multiple classes of devices (e.g., treadmills and bicycles) or between fitness machines from similar device classes that comprise different designs or are from different manufacturers. The result is that fitness devices can work equally well with software applications no matter the device class, model, or manufacturer. From a developer's perspective, the Exercise Device Equivalence Interface 98 simplifies the creation of multi-player games or online fitness training environments in which many users can participate simultaneously, regardless of what model or brand of fitness machines they're using. In one example, a fitness coach can manage an online spinning class in which a dozen or more exercise enthusiasts that are each participating from home or from their local gyms using their own brand and model of exercise bike. In another example, a fitness coach could manage an online triathlon training class in which half the users are working out on treadmills while the other half ride spinning bikes. In these examples, the coaches manage each user's real-time workout progress using a web console application that displays each user's information numerically, graphically, or via live video. This information facilitates the online coach's ability to give targeted performance feedback and
encouragement to each remote user. Because the web console application utilizes the IXExercise Device Equivalence Interface 98, the coach enjoys these benefits regardless of the brand, model or device class of the fitness machine.

[0145] From the perspective of a fitness machine manufacturer, software developer, or entertainment content company, the IXExercise Device Equivalence Interface 98, and the HRG API in general, provides the lynchpin enabling the creation of large, unified audiences for real-time fitness entertainment solutions that are delivered during the user’s workout. Using the IXExercise Device Equivalence Interface 98 feature of the HRG API, a software developer can create one version of a video game that is compatible across multiple brands and models of fitness machines. Alternatively this feature enables fitness machine manufacturers to easily add software and Internet-enabled features to their products by partnering with software and technology companies whose products are IXExercise compliant.

[0146] Many proprietary fitness-entertainment solutions are on the market today, but the proprietary development approach favors the very largest competitors such as Nike’s NikePlus/iPod solution, or the Nintendo Wii’s video game system that incorporates user movement. The true market potential for fitness-entertainment solutions will not be realized, however, until a common development platform is made available to facilitate ad hoc product development collaboration between fitness machines companies, software developers, and entertainment companies. Only HRG offers such a development platform, based on the IXExercise Device Equivalence Interface 98 contained in the HRG API.

[0147] From the user’s perspective, fitness machines represent one of the last consumer products that is not yet connected to the Internet in a standard fashion. The lack of Internet-enabled features is limiting the growth of the fitness machine market and causing its relevance to decline among young media-savvy consumers. Consumers are also suffering from higher rates of obesity and related chronic disease as a result of increased time spent in sedentary media-consumption activities via television, music players, and the Internet. The IXExercise Device Equivalence Interface facilitates the creation of multi-user gaming and fitness training environments that blend entertainment time with exercise time and make working out more social and fun.

[0148] If the services of the IXExercise Device Equivalence Interface 98 are not needed by the application, communication can be handled directly between the IXExercise Treadmill Device Interface and the IXExercise Physics Interface 96, or directly between the IXExercise Bicycle Device Interface and the IXExercise Intensity Interface 97. Alternatively, if the application uses the services of the IXExercise Device Equivalence Interface 98, then this interface can manage communication between the IXExercise Treadmill Device Interface 100 and the IXExercise Physics Interface or between the IXExercise Bicycle Device Interface 101 and the IXExercise Intensity Interface 97.

[0149] FIG. 17 schematically depicts how the HRG particular embodiments combine fitness 106, entertainment 105, to provide a user with many features and opportunities 107. The features and opportunities 107 include access to the Internet to receive coaching instructions from various web portals and support from friends in virtual communities.

[0150] FIG. 18 compares products 113 and features 112. The particular embodiment is the only product that as all of the features as shown in the table. The HRG particular embodiments include standard interfaces with third party content providers, access to mass markets and elite markets, head-to-head user competition, integrations with VCR players, DVD players, and CD-MP3 players, integration with personal computers and dedicated gaming consoles, and integration with multiple gaming brands.

[0151] FIG. 19 is a table that shows the IXExercise features 120 and that the particular embodiment outperforms over others 121. The others lack IXExercise features of being able to allow automatic uploading of exercise data, enabling of true interactive gaming, head-to-head competition, the ability to connect to gaming consoles and personal computers, and to be adaptable to multiple brands of exercise and gaming equipment.

[0152] FIG. 20 shows the platform and interconnectivity of the particular embodiment and shows a piece of fitness related equipment 134 attached with an IXExercise Hardware Interface similar to that illustrated in FIG. 5. The IXExercise Hardware Interface includes firmware and a generic connector and USB chipset. The IXExercise Hardware Interface is attached via a USB or equivalent functioning cable 118 to a personal computer or dedicated gaming console 135. In this particular embodiment, the gaming console or PC 135 communicates with an IXExercise enabled application via a USB driver and IXExercise APIs. Video and/or audio output is then directed to an audio-visual device 137 via an AV cable 136.

[0153] FIG. 21 schematically depicts data communications features of the HRG interfaces. Exercise equipment or hardware communicates via HRG interfaces to interact with services, entertainment software, and entertainment consoles.

[0154] FIG. 22 schematically depicts the more numerous features delivered to users via HRG interfaces. Exercise equipment or hardware communicates via HRG interfaces to receive entertainment software downloads, communication with user communities from social networks both locally and via the Internet, and team play and competition between online local and Internet user communities. This communication may take place using real-time exercise information or stored summary exercise information. The team play and competition may also be from recorded events from local and Internet communities that are downloadable by the local competitor. Other features include receiving and presenting health data storage and analyses, care provider support, and receiving customized health and fitness content. Yet another feature provided by the HRG interface is communicating offers to local members or users of the HRG interface that are customized according to the user’s fitness profile and exercise objectives.

[0155] FIG. 23 schematically depicts the services available from HRG research and development in providing an integrated exercise and entertainment experience. HRG R & D services facilitate the development of collaborative product solutions that combine exercise physiology, consumer electronics, video game and other software development, entertainment & social-networking content, and exercise equipment design to accommodate entertainment based exercising games employing motion attributable to an exercising user, including translation, rotation, sliding, stepping, grasping, lifting, squeezing, pulling, stretching, pushing, and jumping. HRG R & D further enables the related incorporation and designing of efficient data transfer protocols, wireless device designs, and Internet services provisioning. Other HRG R & D integrates cognitive and physical ergonomics considerations into the designs of exercise equipment to accommodate
the integration between entertainment games and the motions
generated by the exercising user.

FIG. 24 schematically depicts the HRG interfaces,
online infrastructure, and R & D services within the overall
HRG business context. HRG provides a comprehensive offering
of technology and services that facilitates the collaborative
development of entertainment-driven exercise solutions
by fitness companies, technology companies, and entertain-
ment content companies. Integration includes incorporating
HRG R & D with standardized interfaces, online infras-
tructure, business and marketing partnerships, and coordinated
consumer branding strategies.

FIG. 25 schematically depicts the communication
protocol options between HRG or third-party Internet Ser-
vices and fitness machine interfaces with entertainment con-
soles via HRG’s application programming interface.
Between HRG or third-party Internet Services and the local
HRG API, various Internet Protocols and communication
methods are available. The IP and communication protocols
include CDMA (code division multiple access), GPRS (gen-
eral packet radio service), EDGE (enhanced data for global
evolution), and XML (extensible markup language). Within
HRG or third-party Internet Services Network infrastructure
communicates with Content Providers via a variety of proto-
cols including XML and VoIP (voice over internet protocol).
Communication between the fitness machine hardware and the
entertainment consoles include USB and Bluetooth.

Data I/O formats for information going to or from the fitness
machine include CSAFE (communications specification for
fitness equipment), ANT wireless sensor network, and fitness
machine manufacturers’ proprietary protocols described
above and delineated in the Appendices. Informational con-
tent conveyed from the hardware to the entertainment con-
soles may then undergo data management integration with
video games and user interaction clients via the HRG API.

While the particular and various alternate embodiments
of the invention have been illustrated and described, as
noted above, many changes can be made without departing
from the spirit and scope of the invention. By way of example,
not limitation, while a USB interface is described for
some embodiments, it should be understood that throughout
the detailed description, any type of interface may be used
without departing from the spirit of the invention. For example,
Blue Tooth, FireWire, or any other custom or off the
shelf type of interface. Similarly, any type of fitness or exer-
cise equipment or computer hardware or software may be
used, or any type of electronic game. Accordingly, the scope
of the invention is not limited by the disclosure of the partic-
ular embodiment. Instead, the invention should be determined
entirely by reference to the claims that follow.

The embodiments of the invention in which an exclusive
property or privilege is claimed are defined as follows:

1. A computer readable medium having executable instruc-
tions to for a method to play an interactive game presentable
on a display and associated with an exercise device operated
by a user, the method comprising:

- installing an application programmable interface (API)
capable of presenting gaming content of the interactive
game modifiable by the conveyed motion generated by
the user operating the exercise device, the API config-
ured for receiving signals proportional to the conveyed motion;
associating the signals to at least one symbolic representa-
tion of the interactive game;

- processing the signals in proportion to parameters
defined by the gaming content; and

- changing the status of the symbolic representation
within the interactive game in proportion to the signal
processing.

2. The computer readable medium of claim 1, wherein
receiving signals proportional to the conveyed motion
includes motion associated with translation, rotation, sliding,
stepping, grasping, lifting, squeezing, pulling, stretching,
pushing, and jumping.

3. The computer readable medium of claim 1, wherein the
API includes at least one of a physics interface, an intensity
interface, a device equivalence interface, a treadmill device
interface, a bicycle device interface, and a generic device
interface interposed between an application program present-
ing the gaming content and the exercise device.

4. The computer readable medium of claim 3, wherein the
device equivalence interface allows communication between
gaming applications and the exercise device.

5. The computer readable medium of claim 1, wherein
associating the signals to at least one symbolic representation
includes a mechanical depiction, a life form depiction, and a
graphic icon.

6. The computer readable medium of claim 5, wherein the
mechanical depiction includes automobiles, ships, airplanes,
spacecraft, submarines, and sporting equipment.

7. The computer readable medium of claim 6, wherein the
sporting equipment includes baseball, football, hunting, and
fencing.

8. The computer readable medium of claim 5, wherein the
life form depiction includes animal, vegetable, human,
microbial, and imaginary.

9. The computer readable medium of claim 5, wherein the
graphic icon depiction includes arcade game symbols.

10. The computer readable medium of claim 1, wherein
processing the signals in proportion to parameters defined by
the gaming content includes self-competitive and inter-per-
sonal competitive games.

11. The computer readable medium of claim 5, wherein
changing the status of the symbolic representation includes
changing at least one of the appearance and location of the
mechanical depiction, the life form depiction, and the graphic
icon within the gaming content of the interactive game.

12. The computer readable medium of claim 5, wherein the
API is capable of reading a computer readable medium hav-
ing code for an interactive exercise program.

13. The computer readable medium of claim 12, wherein
the API is configurable to present a video overlay, the video
overlay containing at least one of a current heart rate, current
exercise intensity, upcoming changes, time and distance.

14. A method for using an user interactive exercise system
in signal communication with a computer, the method com-
prising:

- installing an application programming interface (API) in
communication with an exercise apparatus and a gaming
application executable on the computer;

- operating the exercise apparatus;

- capturing motion-related data from the exercise
apparatus;

- transmitting the motion-related data to a device capable of
displaying a gaming scenario having a symbolic representa-
tion; and

- manipulating the graphic representation in proportion to
the captured motion-related data.
15. The method of claim 14, wherein capturing motion-related data includes motion associated with translation, rotation, sliding, stepping, grasping, lifting, squeezing, pulling, stretching, pushing, and jumping.

16. The method of claim 15, wherein installing the API includes at least one of a physics interface, an intensity interface, a device equivalence interface, a treadmill device interface, a bicycle device interface, and a generic device interface interposed between an application program presenting the gaming content and the exercise apparatus.

17. The method of claim 16, wherein the device equivalence interface allows communication between gaming applications and the exercise apparatus.

18. The method of claim 14, wherein the symbolic representation includes a mechanical depiction, a life form depiction, and a graphic icon depiction.

19. The method of claim 18, wherein the mechanical depiction includes automobiles, ships, airplanes, spacecraft, submarines, and sports equipment.

20. The method of claim 18, wherein the life form depiction includes animal, vegetable, human, microbial, and imaginary.

21. The method of claim 18, wherein the graphic icon depiction includes arcade game symbols.

22. The method of claim 18, wherein manipulating the graphic representation includes changing the status of the symbolic representation includes changing at least one of the appearance and location of the mechanical depiction, the life form depiction, and the graphic icon within the gaming scenario.

23. The method of claim 18, wherein changing the appearance and location includes within the gaming scenario having a self-competitive content.

24. The method of claim 18, wherein changing the appearance and location includes within the gaming scenario having an interpersonal-competitive content.

* * * * *