# An App, a Marathon, and an Astronaut

*How SmartFoxServer is powering a mobile app on the International Space Station*

## Introduction

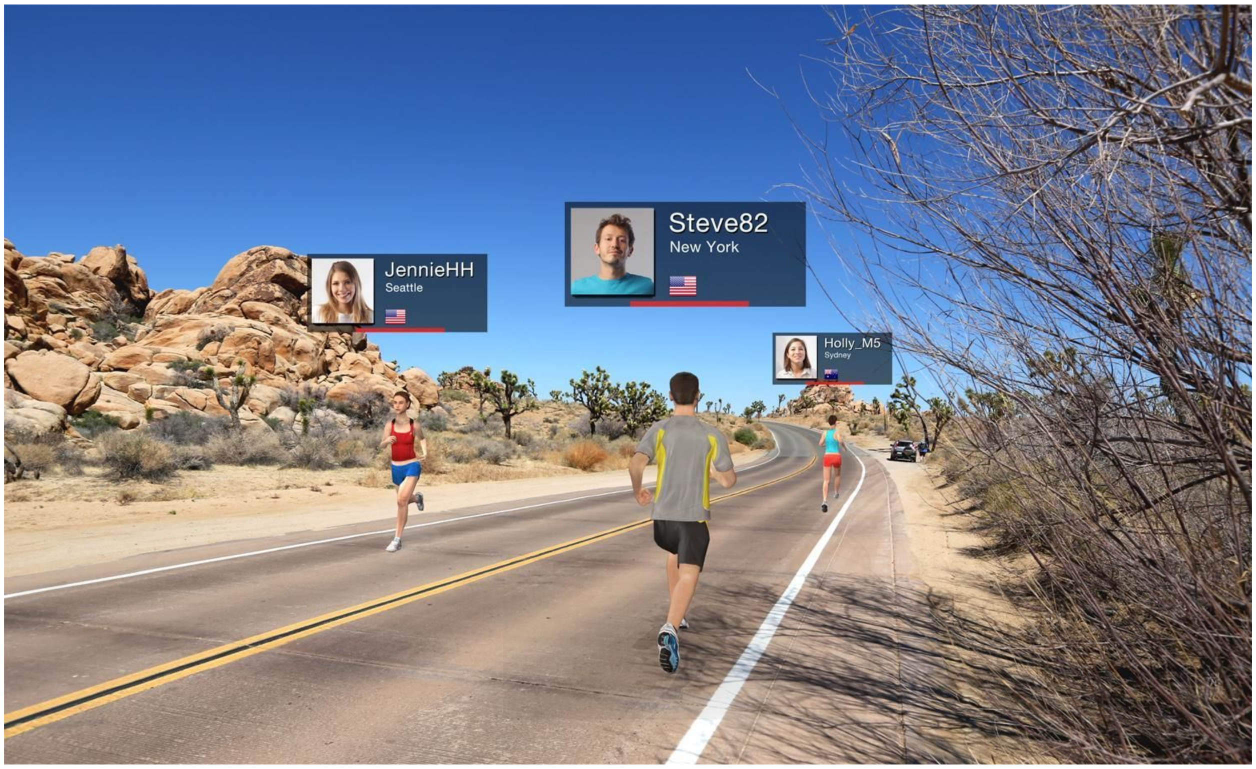
Over the course of a career, there are always a handful of projects that stick out in one's mind. You may spend years producing great work, however, often these projects fade with time into distant memory. This is especially true in the age of mobile and social. Our constant output into the evolving technical and digital ecosystems is truly at historic proportions. If you're lucky enough to be involved in a project where everything - the concept, the client, the technology, the people - seems to align, you know that the output has the potential to make an impact and avoid the fate of distant memory.

Over the last few years, a mobile app called *RunSocial* has been under development featuring some innovative technology and opening up possibilities for consumers and business alike. Recently, the app was approved by the National Aeronautics and Space Administration (NASA) and the European Space Agency (ESA) for use on the International Space Station (ISS) as it provides numerous benefits to the agencies and astronauts.

## About RunSocial

RunSocial is described as a Digital Fitness Revolution. Developed as a mobile app, RunSocial can be installed on tablets and securely attached to a treadmill. As the treadmill is used, the app displays HD video routes from all over the world and synchronizes with the speed at which the treadmill is operating. Using real-time video interpolation, the video route is advanced through as slow or as fast as the treadmill is moving with no discernible degradation in video quality.

Additionally, what differentiates the user experience is the fact that the entire application is multi-user. As you run, you can see other people running the same route from the comfort of their own homes or gyms all over the world.



As of this writing, Apple iOS devices are supported with other platforms in the works.

Consumers can download the app from the AppStore for daily use. Events can be held for larger groups such as corporate clients who can organize events for employee team building or charitable causes. Routes can be run by a team as a relay, making long routes more accessible.

Additionally, RunSocial is now being used by major marathons to bring the experience to individuals who either didn't qualify or are unable to physically attend the event.

RunSocial is comprised of three applications, the client, the server, and a leaderboard application for events.

## About the project

In early 2015, the team at RunSocial began courting NASA. Astronauts are required to exercise in space. Microgravity can cause extreme muscle deterioration over long periods of time and astronauts who spend months on the ISS work to limit those devastating effects. NASA astronaut Mike Hopkins goes into detail what their regiment looks like.

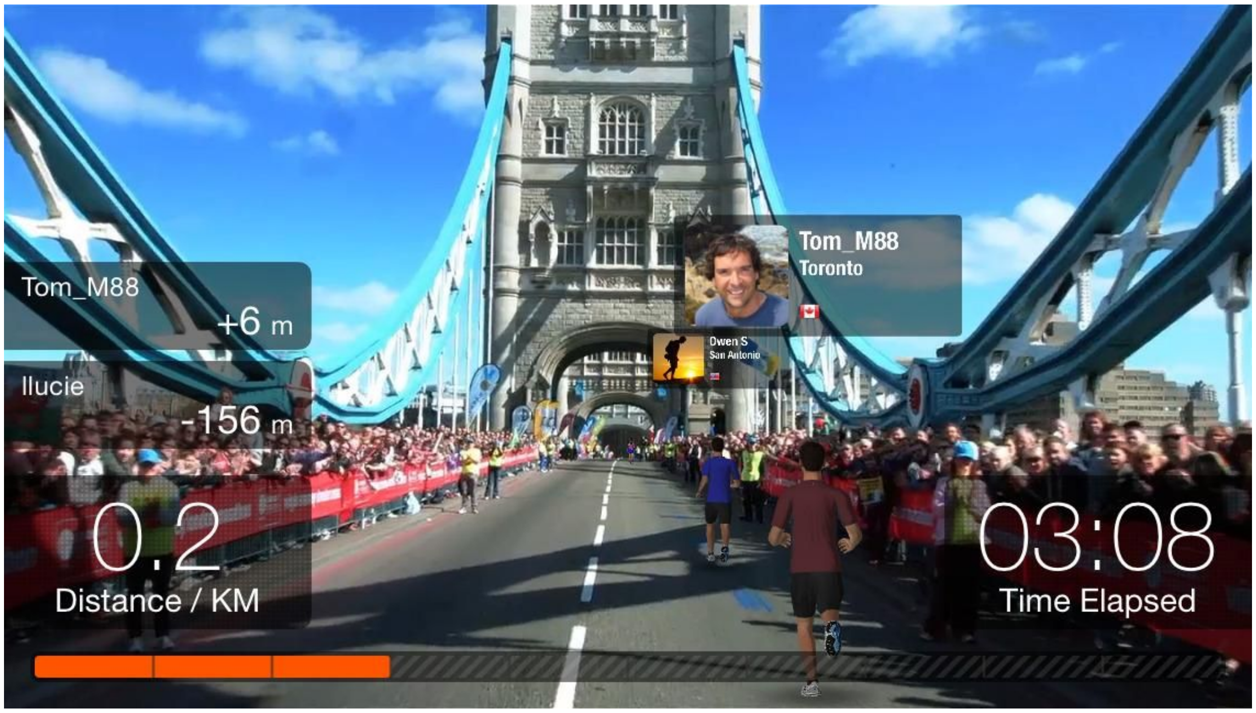
The ISS is equipped with a Combined Operational Load Bearing External Resistance Treadmill (COLBERT) and each individual astronaut has an exercise routine designed specifically for them. As many treadmill users can attest to, this is often rather boring.

In the Tranquility node of the International Space Station, European Space Agency astronaut Luca Parmitano exercises on the Combined Operational Load Bearing External Resistance Treadmill (COLBERT), technically named the Treadmill 2 and abbreviated as T2 (credit: Wikimedia.org)

Additionally, when astronauts spend extended periods of time away from Earth, they can experience isolation stresses.

By providing a social experience and a visually stimulating run, RunSocial delivers a solution to these issues for NASA and the ESA.

Back on Earth, the Virgin Money London Marathon decided to use RunSocial for their event this year to allow people globally to run the marathon alongside runners in real time.



A screen capture from the Digital Virgin Money London Marathon route in RunSocial

There's no better way to promote exercise, NASA, the ESA, and the London Marathon than to have an astronaut run the marathon from space.

British ESA Astronaut Timothy Peake, ran this year's Virgin Money London Marathon from space aboard the ISS.

## Requirements and planning

RunSocial is conceptually a simple application. It reads data from a treadmill and updates the server with a set of variables. Using SmartFoxServer, that data is then distributed to other users running the same route.

Route location and speed are obviously the most frequently updated variables. However, there are quite a few other variables that describe a user's gender, shirt, shorts, location, and username that are used to display runners on screen.

We also need to keep track of relay team members as opposed to individual runners and provide the ability for gym chains to be able to locate and group runners using their facilities.

For this particular event, Tim Peake needed to be visible to all runners and global concurrent users had the potential of being very high.

In all, there's a substantial amount of data to store and distribute to connected users and SmartFoxServer offers us a robust platform to efficiently manage all the tasks.

## Hurdles

Nearly all multi-user applications and games have technical or usability problems that need solving. RunSocial is no different.

### Avatar congestion

A common issue with multi-user scenarios is avatar congestion. As a mobile app, RunSocial runs on relatively small screens with limited real estate. Having 200 runners appear in the viewport simultaneously would diminish the user experience dramatically. Visibility would be obscured and one of the main experiential features of the app - beautiful HD routes, would be hidden.

Since the Digital Marathon coincided with the real-life marathon, all runners and teams started the race simultaneously. As the race progressed, crowds of runners begin to thin as speeds differ. If two hundred is too much, then a hundred thousand is simply not possible.

### VIP visibility

A special avatar was created for Major Peake. It is distinct from all other avatars and can be seen by all runners along the route. This required replication of Major Peake's data for global distribution.

### Message reduction

Since the target concurrent users is quite high, we needed a way to prevent unnecessary and redundant messages being distributed.

### Variable and unstable connections

As a mobile app, connections speeds can vary from device to device and depend heavily on location and carrier technology. Battery power tends to deplete relatively quickly when running resource intensive applications and running a marathon takes time - at least for most of us. This may cause drop-outs where reconnection and event resumption is required.

### Server capacity and failover

When supporting this number of concurrent users, we need an architecture that distributes users among numerous servers while delivering a seamless experience. Additionally, in case of catastrophic failure, failover is critical.

Virtualization and cloud computing has made these tasks far easier when considering infrastructure, however, at the application architecture level with thousands of data messages delivered per second to persistent socket connections, preparing for traffic bursts, capacity and system failures - even if never utilized, is required.

### Logging

Making business decisions is difficult, so any data that we can collect to help illuminate usage patterns, bottlenecks, redundancies, and general analytics is critical. Accessing and writing to any third party data store often is a bottleneck in multi-user applications where response times need to be fractions of a second. Logging data and storing race results for later retrieval needed to be accomplished in an efficient manner.

## SmartFoxServer to the rescue

SmartFoxServer offers a mature and robust solution for multi-user applications and games. Out of the box, SFS provides a wide range of options to store, organize, and distribute users and data.

Additionally, SFS provides a structured server side extension architecture that provides teams with the tools necessary to build high performance and scalable messaging systems. If there’s a need to customize logic, the task is easily accomplished.

Multi-user systems require highly efficient systems to manage data. Each message sent to the server is potentially turned around and distributed to thousands of users. In our scenario, for example, we needed to distribute profile and position updates from Major Peake to every connected user participating in the event.

Knowing how to direct messages is critical to efficiently running the platform. That means distributing users effectively and storing various pieces of data in accessible and secure locations without adding too much overhead to individual users.

### Events, routes, rooms, and runners

User experience is always a priority in any application and RunSocial is no different. Client screen sizes vary, and even the largest has limitations. We can only stuff so many runners into a single view without obstructing the view of the actual route and having runners run on top of each other. Avatar congestion is a serious issue that if ignored, would undoubtedly result in a poor experience for runners.

Most multi-user applications distribute users in a number of ways -whether it be through segregating users onto different servers or limiting access to certain areas of a game. In the case of RunSocial, we have a unique situation in that our routes are quite long. Within the same route, a runner can be located near the beginning and another at the 10th Kilometer. These two runners would not be visible to each other in the real world, and the same should apply in the app.

SmartFoxServer offers a sensible structure for organizing application logic. The top most level is the Zone. Each Zone can have custom Extensions and an unlimited number of Rooms. Rooms are logical units to divide up users.

There are a number of different types of Rooms that support different functionality for different situations, however the Room is the standard organizational unit with which we segregate users.

To support thousands of simultaneous Users, each route in RunSocial will required an unlimited number of dynamically generated Rooms. When a new User logs in, we determine what Route they have chosen, and assign them into an existing Room based on the Event they are participating in or a more generic Route Room where no events are taking place.

But first, we determine the Room capacity. In our scenario, Room capacity is based not only on the number of Users, but also where those Users are along the route. If there are 5 Users at the first kilometer, we determine that there is enough space to insert another avatar at that point in the route. To this particular User, it doesn’t matter if there are 500 other users on the same Route  and in the same Room 10 kilometers ahead - if they are not within a certain proximity, we do not calculate them as visible Users. In this way, a single Room can accommodate hundreds of Users and it’s capacity is determined by route distance.

## A nice hierarchy of data storage solutions

SmartFoxServer offers a nice hierarchy of storage solutions. At the lowest level, data can be stored on the Session using the *Session.setProperty()* method. Moving up the chain, properties can similarly be set on the User or the Room in which they exist. Neither Session properties nor User and Room properties are used by the server in the event model. Avoiding events being thrown can often come in handy when we need to determine certain characteristics about a User or Room from the server side.

Another storage option is User or Room Variables. These variables are eligible for events, so when values are changed, we can notify users in the same Room. The flexibility of User and Room Variables allows us to control which variables are visible and which ones throw events.

Even when creating a Room, we are able to define which events we want enabled on the Room object giving us even more control.

For example, Major Peake, needed to join every single Room for the marathon event enabling all connected Users to receive the data that he transfers to the application. To facilitate this, each Room has a number of properties that define the route and the event, so at the time that Major Peake enters the race, we can cycle through the Room List and determine which Rooms he needs to join.

As other users enter the race, Room are created dynamically to accommodate them. Major Peake needed to join these new Room as well. As the Rooms are created, we are able to broadcast an event that informs Major Peake of the new Room's existence and prompts his client to join.

As a standard User in a Room, you would receive data from all other Users within the set proximity. Major Peake however, would be overwhelmed with data since he has joined every Room in the event.

Major Peake's avatar is unique, he is flagged as a VIP and as such, will only receive data from his primary Room - that is, the Room he first joined. To accomplish this, we set a Property on Major Peake's avatar setting the Room ID of his primary Room. When broadcasting data, we can then iterate through a Room's Users to only broadcast to those whose property match the primary Room ID.

## Optimizing messages and choosing the right protocol

With any mobile application, keeping data transfers to a minimum is important. Not only do we want the user experience to be smooth, but we also don't want our users to incur additional charges for data from their carriers. Communicating back and forth should be done with as little data as possible.

One of the features of SmartFoxServer that makes the platform appealing, is it's ability to handle UDP as well as TCP. The UDP protocol is a widely used protocol to transfer data at high speeds. TCP validates that each packet being transmitted is done so successfully. UDP on the other hand, forgoes that verification, allowing for greater capacity and speed.

When the RunSocial app sends position updates, we use UDP. If one of those updates gets dropped -that is, the server never receives it, or the clients joined to a room never receive an update, we can easily recover on the next packet sent.

Avatars in our scenario move at quite a consistent pace, so predicting position can be calculated easily. Furthermore, the iOS team has developed a corrective positioning method for the client, that gradually fixes any miscalculations in position without it being visible to the user.

To further reduce message size, RunSocial utilizes short object keys and transfers only critical data through the server.

### Failover, and uptime guarantees

Ensuring that your application is accessible at all times is critical to any organization. RunSocial is no different and one of the most difficult aspects of using SmartFoxServer is designing a systems with proper failover that enterprise systems traditionally have.

Because SmartFoxServer is not (yet) clusterable, failover presented a real challenge.

RunSocial provided us with a unique opportunity to develop a system where clients would be migrated to a new server if the primary server were to fail.

Again, RunSocial is an app that provides easily predictable avatar positions. The vast majority of treadmill users will set a speed and run consistently for some time. This allows us to move avatars on the client at a remarkably accurate predictive rate if the server should fail.

The client is designed to detect server disconnection and status. If the client determines that the server has gone away, it packages up the current state of the race, and sends that package to a secondary server.

The secondary server will then “recreate” the experience and the client will continue as if nothing had happened.

## Around the globe and beyond

SmartFoxServer has enabled us to build RunSocial an application that performs and scales well, while offering the safety net required of enterprise software. It’s unique feature set contributes to logical game design and facilitates rapid feature implementation and iteration.

Gradually, rollout of functionality will be phased in to accommodate client versioning and a seamless user experience. More info can be found on the RunSocial website at https://www.runsocial.com/.

### About RunSocial

Run through real-world videos of beautiful locations and see, inside your video, avatars of others doing the same in real time from around the world. And they can see you too. Experience

RunSocial's completely new "mixed reality" technology blending HD real-world video, 3D virtual world capability and social connectivity. The app can track your running speed and you can enjoy it either on your iPad screen or via a bigger screen if connected. Learn more at https://www.runsocial.com/.

## About SmartFoxServer

SmartFoxServer is a multi-platform client/server SDK designed to integrate with major web and mobile technologies, enabling developers to rapidly create scalable multi-user experiences.

SmartFoxServer was created with multi-player games in mind providing powerful tools for the creation of turn-based and real-time games, MMOs, virtual communities and much more. There's really no limit to the number of applications that can be created with SmartFoxServer. Learn more at http://smartfoxserver.com/.

## About the author

Wayne Helman is an award winning digital platform strategist, consultant, and producer having led the production of hundreds of web, mobile, and social projects for such clients as SmartFoxServer developers gotoAndPlay(), Yahoo! Canada, Siemens, The Government of Ontario, The Canadian Broadcasting Corporation, CTV, Global TV, Corus Entertainment, Tween Brands, Under Armour and Panini America's NBA , NFL, and NHL AdrenalynXL products. His company, A51 (http://a51integrated.com/), is based in Toronto, Canada.