Development Optimisation Engine

Amit Prakash Ambasta, Abhishek Chhajer,
Arpith Siromoney & Hrishikesh R. Terdalkar[1,2]

The Earth provides enough to satisfy every man's need
but not every man's greed.

We are a group of friends who study at a small research-oriented institute in Madras, CMI, and are passionate about developing software to make a change; change we can *see*. This document is an attempt to describe our dreams and the challenges we hope to solve through the Imagine Cup.

Development in countries such as ours is characterised by a lack of involvement of the common man, at the grass-roots level. While technology has reached the people, it is our belief that we are not yet empowered enough to influence the decisions that affect us on a day-to-day basis. It is these problems, we feel, that technology should help solve--for a brighter India tomorrow.

Throughout this document, we will fall back on three examples of who we are trying to help. Accident-relief, to begin with, is hampered by a lack of awareness of an optimal strategy towards utilisation of existing resources. Mobilising ambulances and health care personnel who are nearest to scene can help improve survival chances. In the same light, quicker emergency notification leads to quicker response times.

Another, though seemingly different, example is the maintenance of critical resources in supply logistics. If a sewer fails to function, or a street light goes dead, or, in a more drastic situation, a terrorist attack wipes out commodity supply to a city, then quick notification to those whose job is to fix things can go a long way in terms of restoration of the supply. Instant notification, however utopian, is unlikely to lead to an improvement unless optimised strategies are used towards the deployment of reserve resources. And finally, agricultural development can be improved with computation based on statistical evidence and supported by customised user feedback. For example, crop rotation can be used to maximise produce output while maintaining soil fertility, when supplemented by continuously modifiable user data for a number of relevant factors. This is where we feel technology, and our dreams, can come together.

1. [amit.prakash.ambasta, abhishek.chhajer, arpith.siro, terdalkar.24889]@gmail.com
2. Amongst other things, we study together at the Chennai Mathematical Institute (CMI)

Our Dream

As we spent long nights discussing the problems we could see around us, it became increasingly apparent that there was a common thread that ran through the seemingly different issues. Increased involvement at the grass-roots level, would only be possible through pervasive technology already available to the people who were affected by the problem, and faster solutions could only be enabled if new ideas were used to help optimise development through mobilisation of resources.

What we are working towards, is a distributed system that uses SMS and web-based XML-requests to alert services that interact with a centralised engine. The server-client model allows for multiple user agents to be utilised in feedback for the engine. The engine can then perform its optimisations on the users' data via plugins that are managed by the Service Manager. Multiple clients interact with the service-clients registered, as well as with the engine.[3]

How technology steps in

A plethora of tools aid us in our quest: notably, we use the .NET framework, SQL Server, Interoperability techniques (JNBridge Pro), XML-based transactions, SMS Server, Silverlight, Windows Presentation Foundation, and of-course, Microsoft Visual Studio as our trusty IDE.

All problems in computer science can be solved
by another level of indirection.

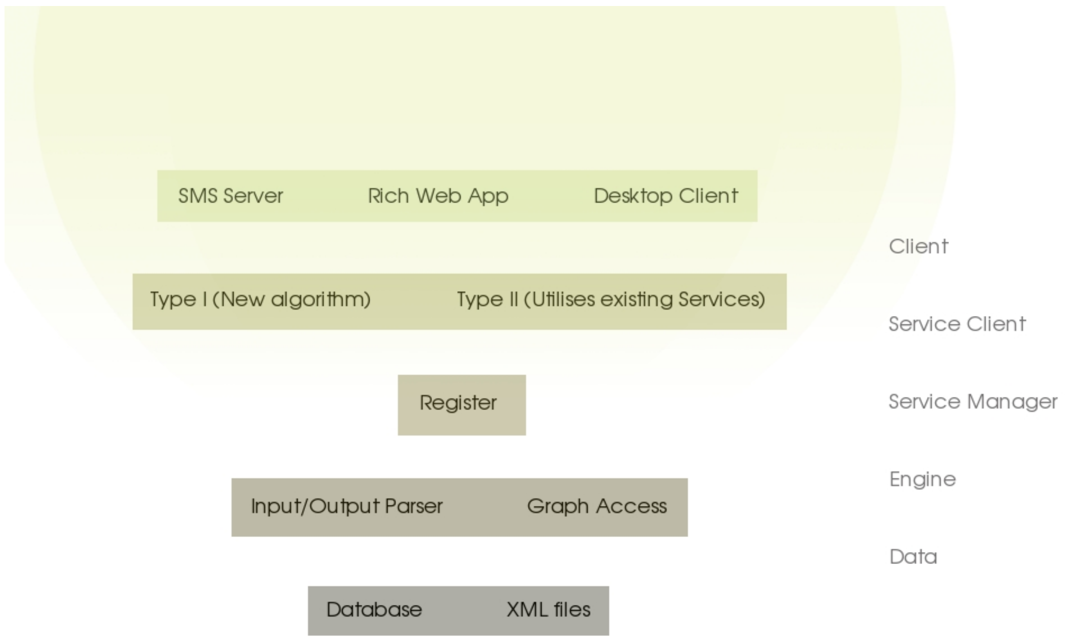
While we certainly can't solve all problems, we are working towards solving the challenges that we are primarily interested in--sustainable development, resource optimisation, and agricultural development--by abstracting them down to the fundamental graph algorithms that we *do* have optimal solutions for. By working on an engine, we believe its extensibility will help increase the potential of developers who want to use software to solve the problems that challenge their customers, clients, and friends.

The first abstraction, therefore, is the fundamental objects that each challenge deals with. For instance, municipal planning authorities have to deal with sewers, garbage disposal units, public transportation, and lighting nodes. On the other hand, hospitals need to keep track of the ambulances they own, the services they can access when they have no solution on their own, and most importantly, the people who need their aid in times of trouble. Objects can be defined either in terms of building blocks (such as nodes and links) or as instances of classes that can be registered with the Object Repository [1].

Of course, buses have to travel from point A to point B, and this is where the ServiceManager [2] comes in. Developers can further extend our software by programming services that utilise the objects stored in the Object Repository. These services can then be run as daemons which, when registered, can be called by the Service Manager when needed. Asynchronous design prevents the engine from having to wait on a service which could have, like our buses, stopped running. Microsoft Silverlight and the Windows Presentation Foundation lend advanced usability to both web and desktop interaction with representations of buses that can be dragged and maps that can be explored.

---

3. We are participating (not yet registered) in the *Design for Development Award* category



The last, and most important, component of our architecture is the Engine [3], that grants services access to manipulate user data. This service oriented model enables the engine to utilise developer designed algorithms that are specific to the problem at hand, while retaining the usability of the plugins that are pre-designed, which solve the challenges we are most interested in.

As described before, SMS and web based interfaces [4] are used to provide alerts to modify nodes in the data that the engine is handling, dynamically. And needless to say, the engine conducts its transactions with the services via XML [0].

Let's take a look at, for example, how our software can be used by health-care providers in rural India. We have developed objects to represent health centers (like hospitals), mobile care providers (like ambulances) and widgets to represent them. The map object enables users to view the routes and positions of the nearest care provider. The web interface is complemented by the SMS service, which allows users to alert the Engine of their location and request assistance. We are developing widgets to represent the objects in Silverlight and several algorithms have already been implemented as services.

A more concrete example lies in optimising agricultural strategies. To maintain soil fertility through crop rotation, a number of factors play a role. Rainfall, soil type, irrigation supply, labour availability, and the value of the crops are all vital to increasing the potential of the farmer's resources. The underlying cost-minimisation graph algorithm has an optimal solution, an algorithm we have coded as a service. Clients can request the Service Manager to start this particular service, which will then perform its computations on that the engine has abstracted to a graph, and has stored. By suggesting a sequence of crops to be planted, taking into consideration the proposed crop distribution for the land and the optimal periods for harvesting, along with the weather dependencies, our Development Optimisation Engine can help farmers increase their land's productivity, and, in a small way, change their lives. The server-client nature of the architecture enables clients as diverse as an SMS Server and a Rich Web Application to be used to input and access data that is being analysed. On the other hand, the inherent distributed model allows for multiple algorithms and sub-routines to be utilised in parallel, increasing efficiency. But most importantly, we feel, is that the structure of the technology that nurtures code development, allowing for developers to work on the part that is key to them, be it a new service-client, or a front-end that is more apt for their purposes. This is what drives us: An easy to use interface, sending a few SMS's, could play a role in helping satisfy those in need. And in our interpretation of Gandhi's words, that is what we are in this world for.

So What?

If we had a few words to describe why we find our work so cool, they would sound something like this: A modular engine with highly usable services that generates widespread inexpensive access to algorithms that solve fundamental problems optimally, can be the bridge between worlds, and help change lives.

We hope we have been able to convey our excitement about our dreams, and our passion for the technology that takes us a step closer to them.