# Towards Personal Process Management

**Thesis A Report**

THE UNIVERSITY OF NEW SOUTH WALES

**School of Computer Science & Engineering**

Supervisor: Helen Hye-Young Paik
Assessor: John Shepherd

Ernest Lie
Prashanthan Ranjan

11 October 2011

## Table of Contents

## Abstract

As business processes continue to be used extensively by large organizations, the importance of considering the individuals involved in these processes is constantly being over looked. In this paper, we propose a system designed to help users manage their personal processes. The system will involve process designers to specify and design personal processes, and end-users, which select and download these processes for personal use. The End-users however will effectively be downloading an interactive to-do list derived from the process, which provides context aware suggestions such as which task is closer or more important in the process. As users continue to use their smartphones to perform most tasks, a mobile-based prototype has also been proposed to implement this system.

# 1. Introduction

Business processes are a set of structured tasks set out to achieve a business goal. Personal processes on the other hand are repetitive day-to-day activities relevant at a personal level. At the moment there are currently many tools that support business process execution, but none for personal processes. This can be attributed to the different nature of personal processes.

Personal processes differ to business processes in the sense that many of the tasks that personal users prefer to do cannot be automated. This means that personal processes are by nature, more manual and are mostly partially automatic in their execution. Personal processes are also in a sense, much more flexible in the order of how we would like to do things. Users may prefer to do things out of sequence while business processes are much more strict and rigorous in how they are defined. These two concepts mean personal processes are very loosely structured, highly variable and that applying the automated tools currently out there for business processes would not be suitable on a personal level.

It is well known in the workflow and business processes area that supporting flexible processes is extremely difficult and is still an open research problem. We recognize that there are different types of personal processes depending on how much control or input the user has over them. For example, a process to cook a meal could vary to great extent between users as users may choose to perform steps in the recipe in a different manner or ignore steps completely. In contrast, a personal process can also involve for example a job application, which users cannot control everything exactly with their own freedom as it is still dependent on the business process that underlines the job application process. These issues need to be considered when designing a personal process management system.

The aim of this project is to create a tool that will allow users to effectively manage personal processes. The tool will convert a personal process to a list of tasks that the user needs to complete to achieve the goal and will help the user in completing these tasks by using context aware suggestions. It will allow users to share and re-use knowledge that is gained from completing personal processes.

# 2. Literature Review

The following literature reviews aim to identify and critique key concepts and ideas posed by others in the area of personal process management. We hope to derive some concepts from these works and improve upon them in the proposed system.

## 2.1 *Personal Workflows: Modeling and Management* (San-Yih Hwang and Ya-Fan Chen), Conference Paper, 2003

San-Yih Hwang and Ya-Fan Chen propose a model for specifying and querying personal processes. It allows for mobile user to specify their own personal processes on the system and then run queries on these personal processes so as to be able to assist them with planning their activities while on the move.

One of the key ideas gained from this paper is the fact that personal workflows can be represented as a set of tasks using a graph data structure. Each node in the graph has associated metadata that will help the system determine interdependencies between tasks.

The paper clearly identifies issues specific to personal process management as opposed to business process management. It clearly identifies that personal processes are related by executable location and time, and that, unlike business processes, users generally do not impose rigid rules to the control flow of these tasks. It goes on to state that the any personal workflow management system should only remind or provide suggestions to the user, and not to impose the execution of the task onto the user as evidenced on business-oriented workflow management systems.

We hope to make use of the principle of using graphs to represent the personal process and improve upon this system by providing context aware recommendations on tasks to the user. This current system only shows a list of tasks to complete to the user and although contains contextual information for each task, it does not go ahead and provide recommendations to the user based on the context.

## 2.2 *Personal Process Management: Design and Execution for End-Users*(Ingo Weber, Hye-Young Paik, Boualem Benatallah, Corren Vorwerk, Liangliang Zheng, and Sungwook Kim), Technical Report, 2010

This paper is the backbone of where most of our ideas come from. It attempts to define what a personal process is, how it differs from business processes, the technologies required to effectively come up with a solution, the architecture and more.

One of the main things we took from this paper is the definition of a personal process. The paper defined that a personal process is a business processes as experienced and described by a single person. It also notes that personal processes differ in business processes in that they are not as automated, and require more flexibility and simplicity. They attempted to create their own personal process modeling language called PPML, which is quite relevant and notes the control structures that it finds more important than others. It also described its architecture and solution: FormSys, an execution program which executes PPML through a variety of tools used externally (JQuery, Acroform, Intalio) and models some scenarios on how they can solve personal processes.

Our project extends from their core idea, but we adjust our approach to a mobile and to-do list environment instead. One of the key ideas from the paper that inspired this is that it stated to cover the interleaved parallel routing control flow pattern (where one user completes the task, but must complete all tasks at least once) a checklist would be appropriate. We will also be using a pre-defined modeling language (YAWL) instead of PPML as PPML does not contain a toolset that can easily generate a description file of a designed process, which we will need to convert to a To-Do list.

# 3. Application Review

This section focuses on current systems, which possess similar functionality to the proposed system. However as they are not thoroughly suited towards personal process management, we hope to combine specific features and improve upon the proposed system.

## 3.1 Remember the milk

Remember The Milk (RTM) is a web-based task management application with an open API that is targeted towards providing the services through as many means as possible. It can therefore be easily integrated into other web, mobile and desktop applications.

One of the key aspects of the application is that it allows the user to specify locations that can then be assigned to tasks so that the user is able to view tasks by location on a map. It also allows the user to specify an estimated completion time for each task along with many forms of context.

Some criticism we offer is that multiple tasks cannot be viewed on the map for comparison, nor does it allow users to create subtasks - showing control flow can be improved.

## 3.2 Checkmark

Checkmark is a to-do list manager app on the android mobile platform. It allows users to create simple lists, group tasks into sub-tasks, group lists into different categories (entertainment/shopping etc.), and even assign simple hyperlinks to applications like Gmail and Google Maps if an email task was required, or a task had a particular location assigned to it. Bonus functionality includes the ability to sync to Google Tasks, Calendar (their own separate Checkmark Calendar) and you can create your own templates.

What we picked up from this app the most are the interface designs, how a page links to another, and the ability to assign a location and email context to a task was certainly useful.

There are some limitations of this app however:

- There is no view for viewing all the tasks with a location simultaneously so you can see which task is closer. You can't see which tasks are closer to a deadline at present.
- A due date can be assigned to a task, but only by a certain date. It would be useful if a user can assign a time deadline as well, or a time frame.

The key idea we would like to extend on this is that users may want to design not only a checklist, but impose certain conditions on which tasks to do first, and which ones are the most convenient

## 4. Proposal

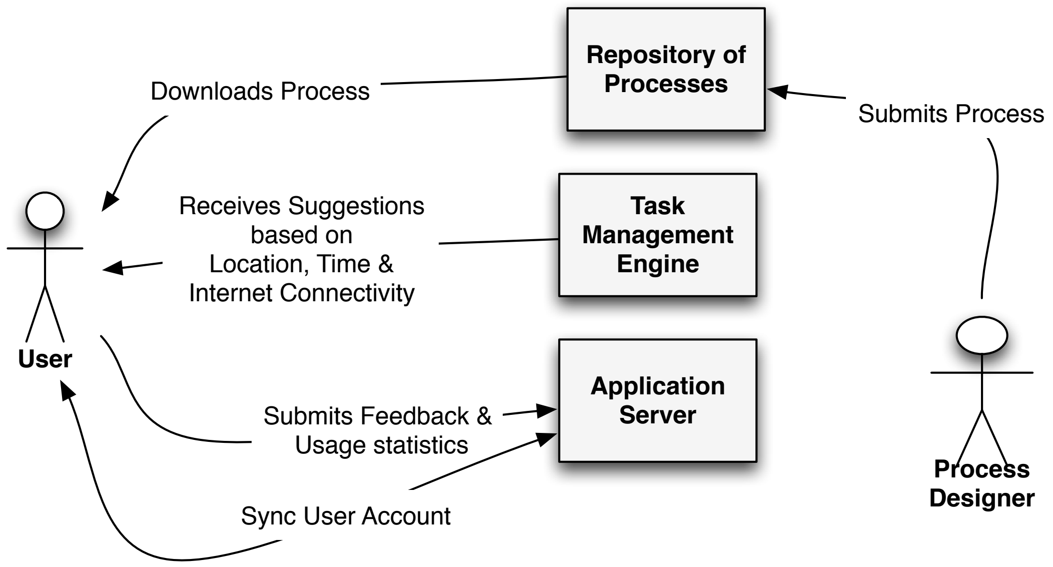


**Figure 1: Proposal Overview**

After considering the current state of personal workflow management systems we have decided to create a to-do list application that can allow individuals to effectively manage their personal processes.

A to-do list was chosen as the management tool of choice as they are suite to personal process because they only involve one user performing one task at a time. It is appropriate for the system as one of the aims of the project is to "allow users to effectively manage their personal processes" and to-do lists are the most popular and familiar tools for personal task management. To-do list have evolved into being shareable and include sort and search functionally which have been deemed to be quite effective.

The system will allow users to download one or more processes from a repository of personal processes. Processes will be submitted to this repository by experienced process designers. The application will present all the personal processes associated with the user in the form of a to-do list. The application will prioritize tasks based on location, due date, time to completion and if the task requires an Internet connection. The user will be able to able to specify the order of priority or the order of the tasks themselves and will be able to select from a number of filters (incomplete tasks, tasks by process etc.) and sorting options to be able to view the task list in their preferred form.

Users will able to sync status of their tasks with the server so as to be able to retrieve their task list and progress in case of data loss on the mobile. The user can send feedback about the processes back to the server. This will be an opt-in option and the data will be collected in the background. Usage statistics such as how long does it take a user to complete a task, and how often does a user stray away from the process control flow will be collected by the application and be used to improve it in further iterations.

## 5. Modeling tools: BPMN vs. YAWL

Business Process Modeling Language (BPMN) and Yet Another Workflow Language (YAWL) are both business process-modeling languages. They are used to model business processes and have an execution engine underneath that attempts to implement the process.

BPMN is well known. It is in fact a standard supported by many well-known software vendors such as IBM, HP and Apple. YAWL is as described, another workflow language, and although not a standard, it covers a huge scope of business processes or workflow patterns that can be created.

We chose YAWL and the reasons for this are:

- We found it is more simple to use, and simple to understand
- YAWL has formal semantics supporting it so that interpretation should be precise and processes verified for implementation
- YAWL is known to have more comprehensive support for different workflow patterns (control flow)
- A BPMN to YAWL plugin is available
- The XML output of YAWL, which is necessary for us to understand for our translation engine into a mobile language, is easier to understand and control. YAWL also has less constructs

Disadvantages:

- Not a standard, an academic project
- The joins and splits code in XML is not very intuitive

Differences:

- BPMN has connector chains (multiple gateway chaining together), while YAWL doesn't. YAWL includes joins and splits in the task objects.
- YAWL requires processes to have only one start and one end condition. In BPMN, multiple start and multiple end events are allowed.

It is important to note that both languages have their own issues - a list of BPMN issues and YAWL issues can be found under the 2 links:

BPMN - http://www.omg.org/bpmn/BPMN_2-0/BPMN_2-0_Issues.htm
YAWL - http://code.google.com/p/yawl/issues/list

As we will be dealing with a single person process, a lot of the issues do not apply to us and so we will not be focusing on their analysis in the scope of our project.

# 6. Design

## 6.1 System architecture

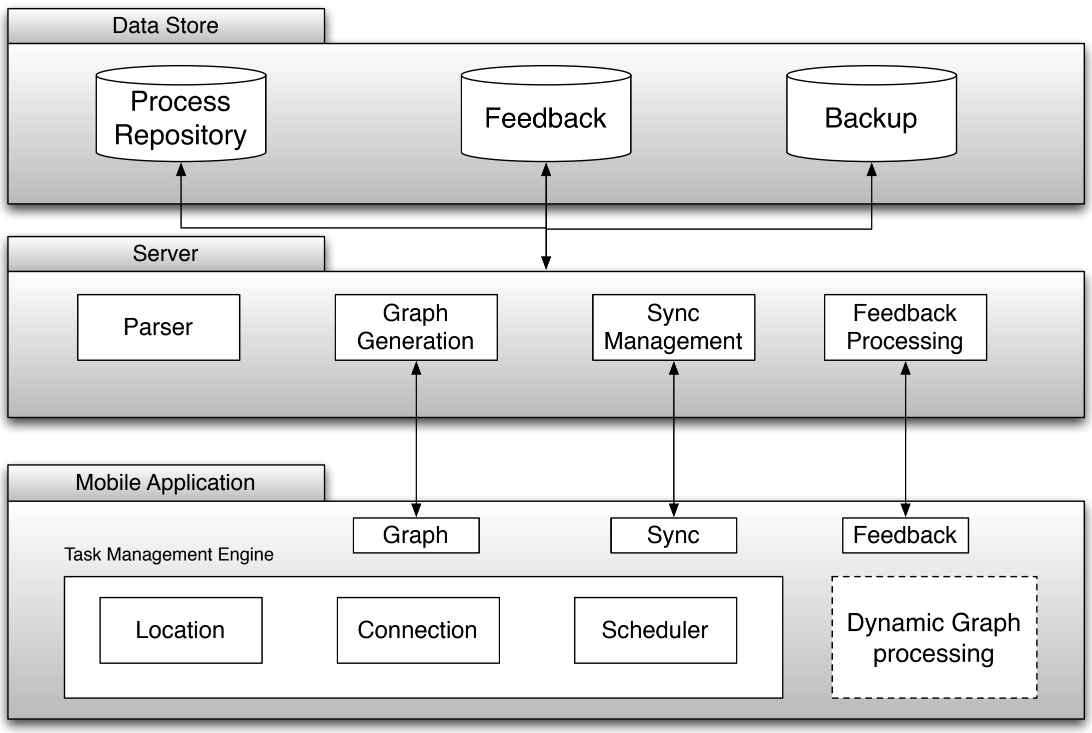


**Figure 2: System Architecture Diagram**

The architecture of the system consists of three main layers: Data Store, Application Server and Mobile Device.

The Data Store comprises of the Process repository, the Feedback database and the Backup database. The Process repository will contain all personal processes that have been submitted by process designers and are available for download by the users. The Feedback database will store suggestions sent in by users and all usage statistics collected by the application. The Backup database will store the user's account details and the status of their processes in order to be able to restore it in the event of data loss.

The Application Server contains a Parser, Graph Generator, Sync Management and Feedback Processing modules. The Parser module will convert the xml representation of the personal process diagram and send it to the Graph Generator module which will then create a graph representing the to-do list view of the process. The Sync Management process is used to transfer the user's account details and the status of the various processes being used by the user to the Backup database and retrieve data from the database if need be. The Feedback Processing module will process all statistics and feedback gained from users of the system and stores the results in the Feedback database.

The Mobile Device application will consist of data access objects which read and send graph, sync and feedback modules to and from the server, and also consists of the Task Management Engine. A Dynamic Graph processing patch may also be included as an extra to handle dynamic processes – for example, a task involves a user to select a car he/she wants to buy, but wants to create a list of possible candidates, and so a subtask will then be added as a requirement to complete this task where he/she needs to create a list of candidates and select a car from the list.

Graph data structures received from the Application Server will be passed on to the Task Management Engine. This engine will then prioritize tasks based on location, the scheduling aspect or the Internet connectivity aspect. The Task Management Engine can be considered as a master class that receives the recommendations from the Location, Scheduler and Internet Connectivity and then combine and provide the best possible recommendation to the user. The Location tasks will recommend tasks that are nearby to the user's current location, while the Scheduler class will recommend tasks based on the availability of the task at that time and the estimated duration of the task and the Internet Connectivity will recommend tasks that require internet connectivity if an internet connection is present. The Feedback Processing and Sync Management modules tie in with their respective modules on the Application Server.

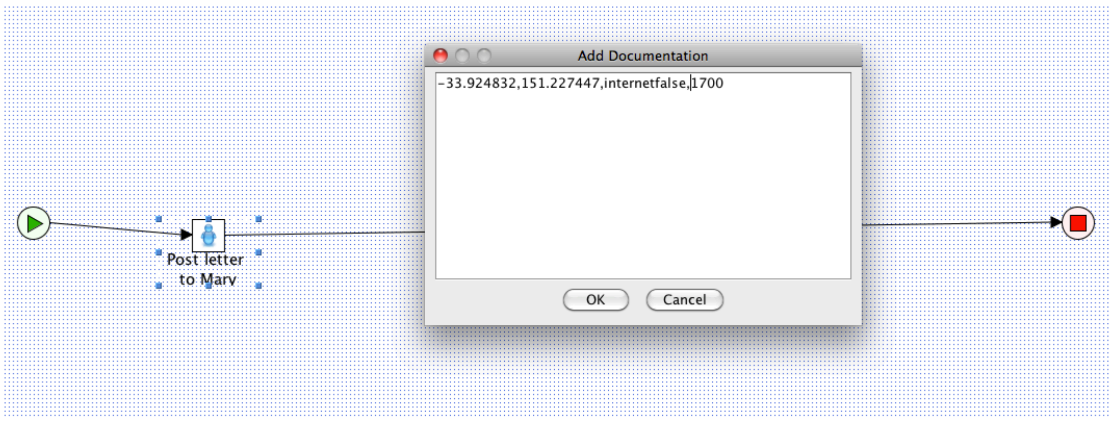
## 6.2 Context of tasks

To allow our system to give personal users a better way of handling personal processes and not just simply marking tasks complete as they go through each process – we can add contextual information for each task such as Time and Location and recommend to the user which task is closest and which task have a time deadline.

We understand that context can be extended and not restricted to just Time and Location for a task, but other contexts such as Temperature of the day or a Rating indicator. The scope of our project will not handle a general context assumption although we understand this can be handled in a less ad-hoc manner. For this project, Time and Location will be the only context available.

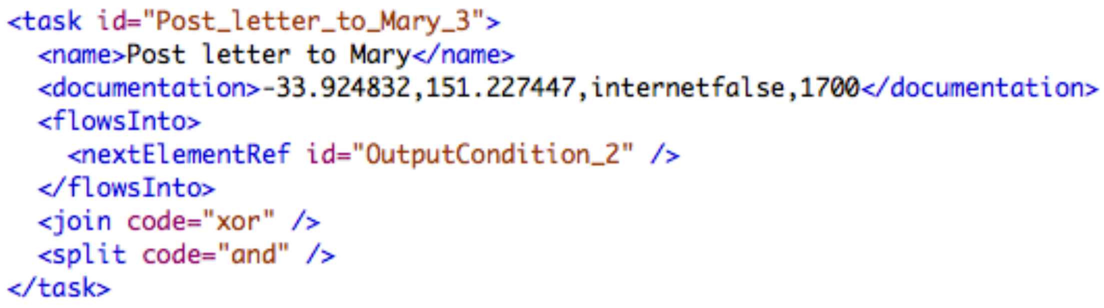
Location will be based on longitude and latitude co-ordinates and time can be based on a time interval, starting & closing time or just a deadline. These will be added via the YAWL designer as a documentation tag.

As an example, we can assign "Post letter to Mary" at Kingsford Post Office and this has to be completed by 5pm(no internet connection required)



```
-33.924832,151.227447,internetfalse,1700
```

**Figure 3: Documentation tag**



```
<task id="Post_letter_to_Mary_3">
  <name>Post letter to Mary</name>
  <documentation>-33.924832,151.227447,internetfalse,1700</documentation>
  <flowsInto>
    <nextElementRef id="OutputCondition_2" />
  </flowsInto>
  <join code="xor" />
  <split code="and" />
</task>
```

**Figure 4: Documentation XML**

## 6.3 Implementation Design

Our proposal requires a process designer to initially use a business process tool (YAWL) to draw the process. This language cannot be translated to Java (the language used for Android mobile) and so we will be designing our own translation engine. The engine will primarily aim to translate a business modeling language into a graph structure. Since most of the control-flow structures in our scope will be rather simple (as we expect personal processes not to be rather complicated), this can be done quite easily.

To do this step, we will take a YAWL file written in XML, and designed by the process designer. The following shows an example of a University Enrolment process as viewed by a personal user:

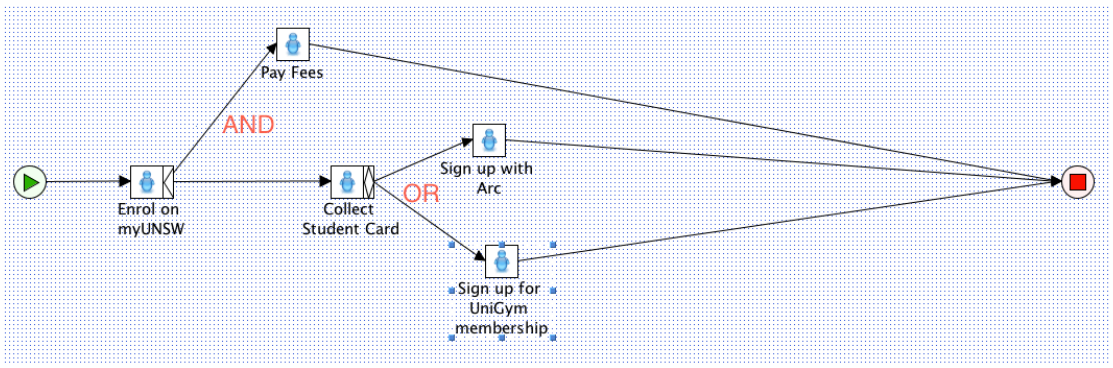
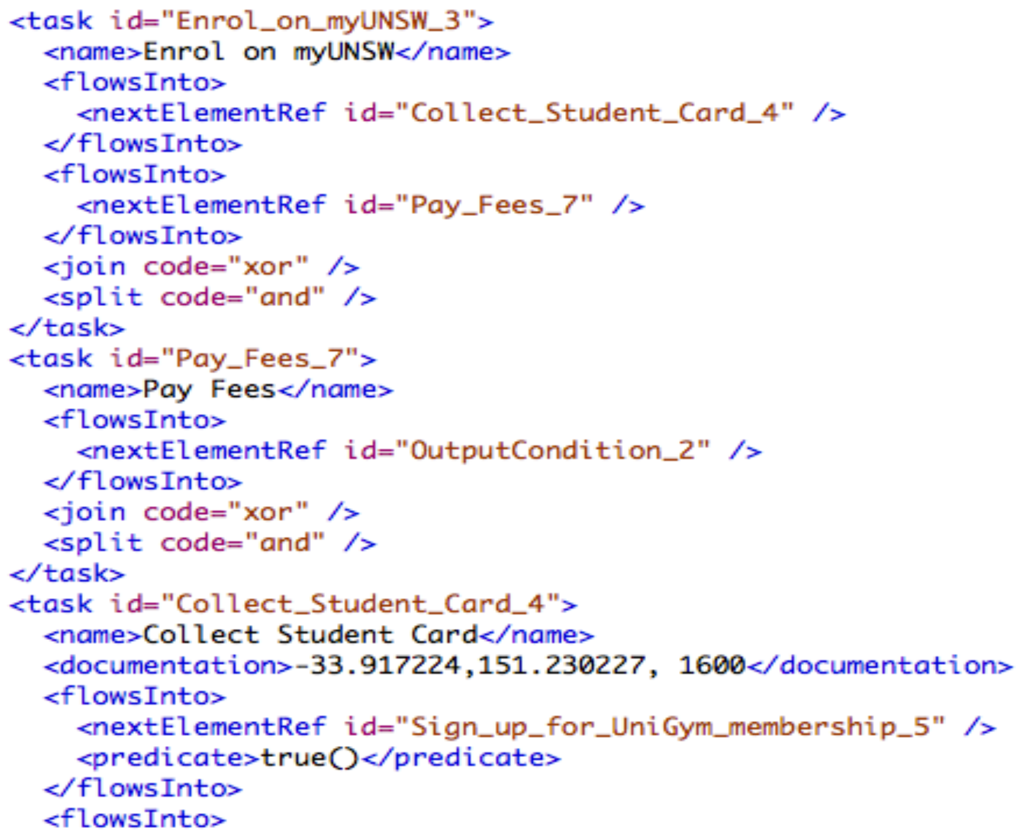


**Figure 5: University of NSW Enrolment Process**

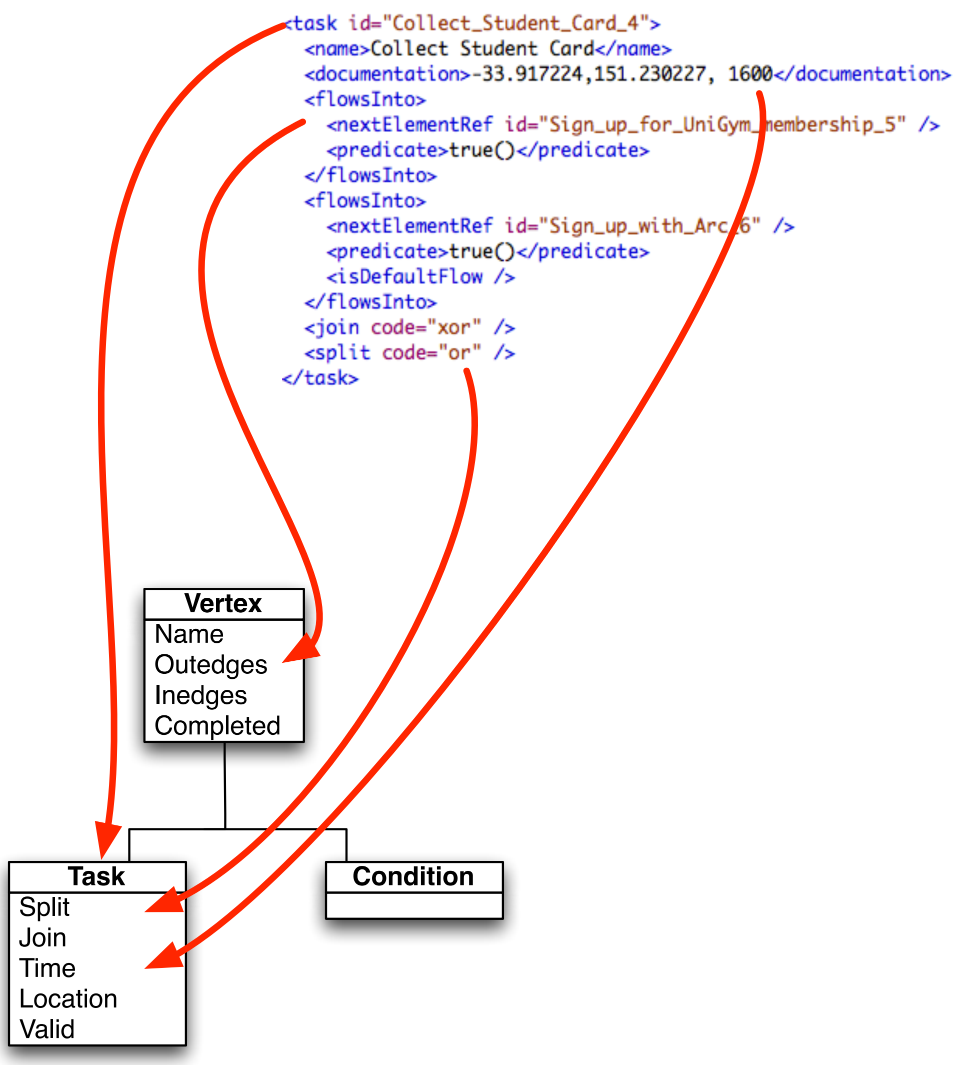


```xml
<task id="Enrol_on_myUNSW_3">
  <name>Enrol on myUNSW</name>
  <flowsInto>
    <nextElementRef id="Collect_Student_Card_4" />
  </flowsInto>
  <flowsInto>
    <nextElementRef id="Pay_Fees_7" />
  </flowsInto>
  <join code="xor" />
  <split code="and" />
</task>
<task id="Pay_Fees_7">
  <name>Pay Fees</name>
  <flowsInto>
    <nextElementRef id="OutputCondition_2" />
  </flowsInto>
  <join code="xor" />
  <split code="and" />
</task>
<task id="Collect_Student_Card_4">
  <name>Collect Student Card</name>
  <documentation>-33.917224,151.230227, 1600</documentation>
  <flowsInto>
    <nextElementRef id="Sign_up_for_UniGym_membership_5" />
    <predicate>true()</predicate>
  </flowsInto>
  <flowsInto>
```

**Figure 6: University of NSW Enrolment XML**

The following translations will then be made to convert it into a graph structure:

- Root node will be the input condition (YAWL only allows for one input condition).
- Tasks will be translated to Vertices (Task subtype)
- Conditions will be translated to Vertices (Condition subtype)
- <flows into> tags translated as Outedges
- Inedges will be derived from <flows into> tags
- Edges with labels (Yes and No for conditions) will be translated into a variable
- Split and join codes will be verified (YAWL split and join codes are not quite intuitive) and added as variables in a Vertex
- Documentation tags will be parsed to Time and Location context variables



```
<task id="Collect_Student_Card_4">
  <name>Collect Student Card</name>
  <documentation>-33.917224,151.230227, 1600</documentation>
  <flowsInto>
    <nextElementRef id="Sign_up_for_UniGym_membership_5" />
    <predicate>true()</predicate>
  </flowsInto>
  <flowsInto>
    <nextElementRef id="Sign_up_with_Arc_6" />
    <predicate>true()</predicate>
    <isDefaultFlow />
  </flowsInto>
  <join code="xor" />
  <split code="or" />
</task>
```

**Figure 7: XML to UML transformation**

### 6.4 Backup system

When users download and use processes through our application, they may additionally backup all their processes on their To-do list onto our server. We really only need to save for each process, a list of vertices and their status code (completed, ignored etc.), instead of the whole graph to minimize the data required to be transmitted

Example:

List<VertexSave> UniEnrolment = EnrolOnMyUNSW, CollectStudentCard, PayFees, SignUpArc, SignUpGym.

EnrolOnMyUNSW.Completed = true

CollectStudentCard.Completed = false
. . .

## 6.5 Conditional Execution

Conditional Execution refers to a step in the process where a user meets a gateway – that is, if a user meets the condition, he/she goes the "yes" path, if a user fails the condition, he/she goes the "no" path.

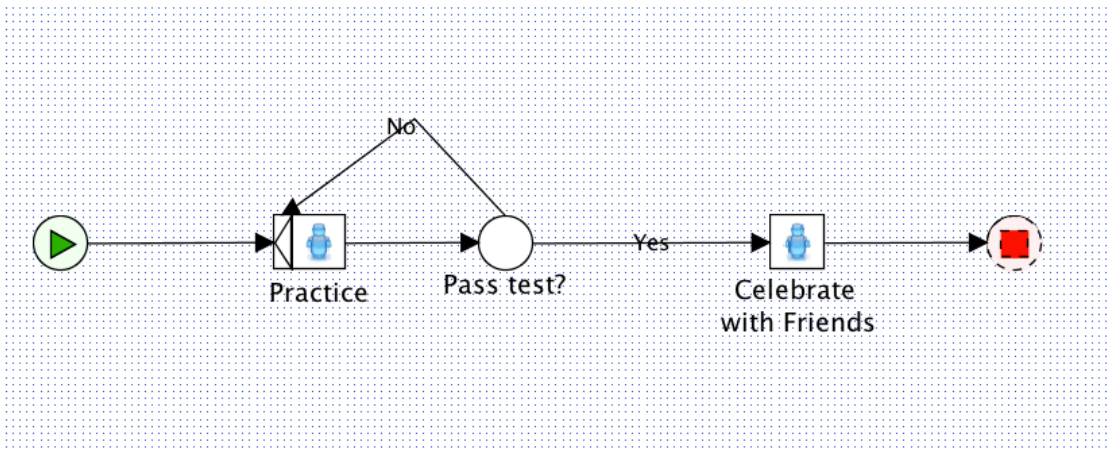
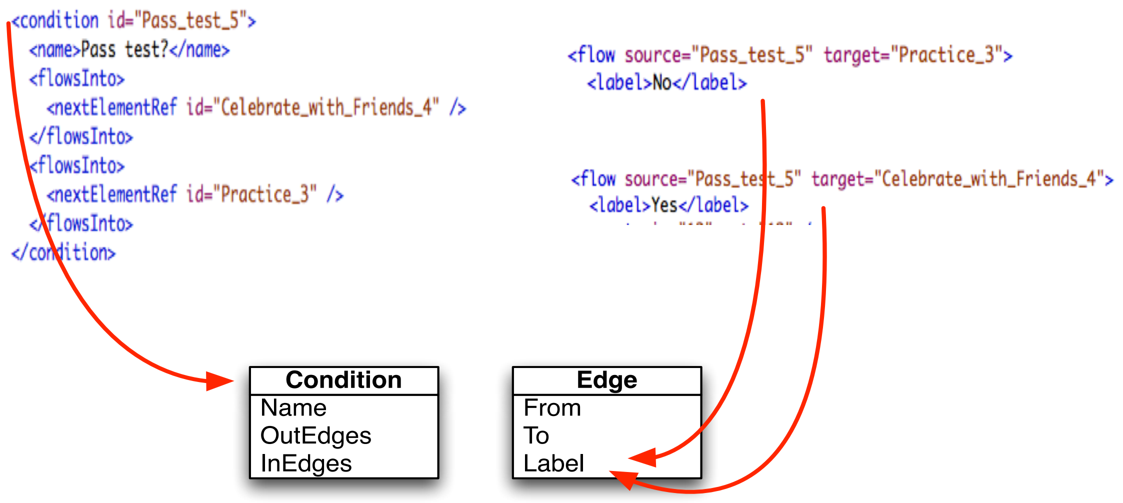


**Figure 8: Conditional Execution Example**

To handle this, we define a Condition as a subtype of a Vertex in a graph (the other subtype being a Task). The Condition label in YAWL (Pass test?) is the condition, and the edge labels in YAWL (Yes & No) are the resulting paths.



```
<condition id="Pass_test_5">
  <name>Pass test?</name>
  <flowsInto>
    <nextElementRef id="Celebrate_with_Friends_4" />
  </flowsInto>
  <flowsInto>
    <nextElementRef id="Practice_3" />
  </flowsInto>
</condition>
```

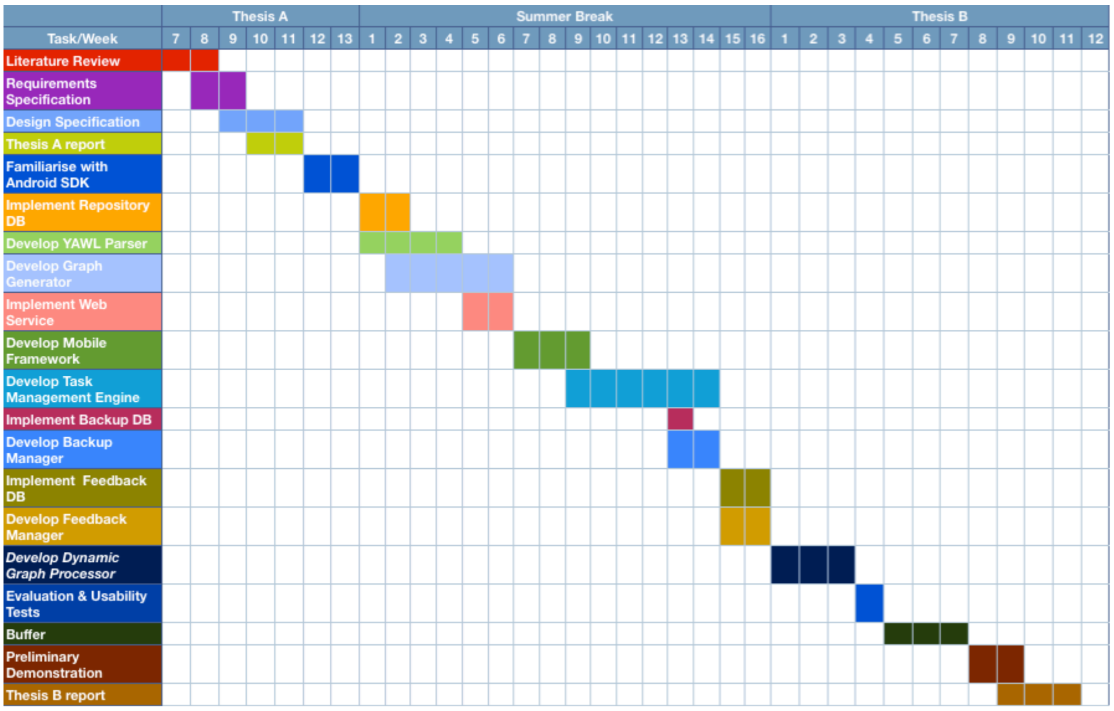
```
<flow source="Pass_test_5" target="Practice_3">
  <label>No</label>

<flow source="Pass_test_5" target="Celebrate_with_Friends_4">
  <label>Yes</label>
```

**Figure 9: Conditional Execution XML to UML**

## 6.6 Dynamic Process Management

Personal processes we have discussed is much more variable and flexible in nature, and so it is very important we allow our application to be flexible for the everyday user. One of the main things we think the end-user would like to do with their tasks and processes is to do whatever they want with it. And so, our system should allow users to add a task on the fly, or remove one, as well as ignoring a task even though it has been defined in a process to complete it before advancing to a next task.

## 7. Schedule



| | Thesis A | | | | | | | Summer Break | | | | | | | | | | | | | | | | Thesis B | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Task/Week | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 | 16 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 |
| Literature Review | ■ | ■ | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| Requirements Specification | | ■ | ■ | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| Design Specification | | | ■ | ■ | ■ | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| Thesis A report | | | | ■ | ■ | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| Familiarise with Android SDK | | | | | | ■ | ■ | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| Implement Repository DB | | | | | | | | ■ | ■ | | | | | | | | | | | | | | | | | | | | | | | | | | |
| Develop YAWL Parser | | | | | | | | ■ | ■ | ■ | ■ | | | | | | | | | | | | | | | | | | | | | | | | |
| Develop Graph Generator | | | | | | | | | ■ | ■ | ■ | ■ | ■ | | | | | | | | | | | | | | | | | | | | | | |
| Implement Web Service | | | | | | | | | | | | ■ | ■ | | | | | | | | | | | | | | | | | | | | | | |
| Develop Mobile Framework | | | | | | | | | | | | | | ■ | ■ | ■ | | | | | | | | | | | | | | | | | | | |
| Develop Task Management Engine | | | | | | | | | | | | | | | | ■ | ■ | ■ | ■ | ■ | ■ | | | | | | | | | | | | | | |
| Implement Backup DB | | | | | | | | | | | | | | | | | | | | ■ | | | | | | | | | | | | | | | |
| Develop Backup Manager | | | | | | | | | | | | | | | | | | | | ■ | ■ | | | | | | | | | | | | | | |
| Implement Feedback DB | | | | | | | | | | | | | | | | | | | | | | ■ | ■ | | | | | | | | | | | | |
| Develop Feedback Manager | | | | | | | | | | | | | | | | | | | | | | ■ | ■ | | | | | | | | | | | | |
| *Develop Dynamic Graph Processor* | | | | | | | | | | | | | | | | | | | | | | | | ■ | ■ | ■ | | | | | | | | | |
| Evaluation & Usability Tests | | | | | | | | | | | | | | | | | | | | | | | | | | | ■ | | | | | | | | |
| Buffer | | | | | | | | | | | | | | | | | | | | | | | | | | | | ■ | ■ | ■ | | | | | |
| Preliminary Demonstration | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | ■ | ■ | | | |
| Thesis B report | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | ■ | ■ | ■ | |

**Figure 10: Schedule Plan**

An agile based software development methodology will be used due to the size of the project and the numerous modules that make up the whole system. The main tasks include familiarization with the mobile SDK, developing the graph generator and the task management engine. The system will be constantly tested as progress is made. A detailed breakdown of tasks is provided in the project schedule.

## 8. Evaluation Plan

Our evaluation of our system aims to deal with the key problems we aim to solve in personal processes. Some of these include:

- Does our system allow users to effectively manage personal processes?
- Does the scope of our system give users enough flexibility?
- Do context aware suggestions align with user preferences?

To solve some of these questions, we carry out a series of tests.

1. Time Benchmark
One of the key questions we want to test is if our system really allows users to effectively manage their processes. An obvious way to test this is through a time benchmark - measuring how long it would take for an end-user to complete a personal process, with our application, and without (control group). We will calculate the average time of completion for each group given a scenario.

2. Prioritisation matching
We want to test if our context-aware suggestions, feedback collection and the whole of idea of processes and task dependencies is effective in allowing users carry out their personal tasks. And so, we will check this by giving end-users scenarios where they are given:

- A location
- A series of tasks
- A map of where the tasks are to be carries out

And ask them which tasks they would like to carry out and in which order. This will help to test if our system's task management suggestions really align with users' preferences

3. Usability tests
Usability tests are important; as effectiveness and flexibility of a system are words we understand as to be quite subjective. Even the interface can be what hinders an effective system, much rather than the features. And so, we will carry out a few usability tests breaking them down to several categories:

- Automation features of our system

  Our system allows for manual and automatic tasks to be carried out, but what mostly differentiates our system to the myriad of To-do list applications available out there is the automatic component. We want to test if an end-user would really use this feature and if it will be useful to them

- Interface Design

  The interface needs to be designed to end-user satisfaction. A usability test asking for feedback on our interface design and how we can make it better will be carried out.

- Flexibility of process control

  The system will include a dynamic extension to control flow and give users the flexibility in controlling the process regardless of what the process is defined to be carried out. This is important, as it is how we define our solution to effectively manage personal processes given their highly variable nature.

4. Data collection
One of the easiest ways to measure the effectiveness of our system is to collect data from end-users as they use our application. Some of the data that would be useful include:

- How many processes does the end-user download on average?
- How often does the end-user use our application?
- How long does it take for a user to complete a certain task?

The way this will be conducted will be to conduct an open research experiment involving UNSW's Orientation week. New students who are seeking to enrol at UNSW will be able to download the app, which will have a pre-defined process and help guide the new students to complete the enrolment process.

Data will be logged, and feedback collected from them will be important in finding out if the application can be effective. Data is only collected via an opt-in basis and only after our experiment is approved by the board of ethics with a formal evaluation plan submitted.

# 9. Conclusion

As the popularity of business processes continues to increase, individuals involved in these processes on a personal scale are being forced to co-operate with these business processes. However at this point in time, there are no effective tools for these individuals to manage the personal processes that they are concerned with. This proposed system presents an effective method to allow individuals to keep track of their personal processes via smart and dynamic task lists in order to be able to complete these processes as efficiently as possible

## Bibliography

Remember The Milk. Remember The Milk. 2011. September 2011 <http://www.rememberthemilk.com/>.

San-Yih, Hwang and Chen Ya-Fan . *Personal Workflows: Modeling and Management*. Kaohsiung: National Sun Yat-sen University, 2003

Ingo Weber, Hye-Young Paik, Boualem Benatallah, Corren Vorwerk, Zifei Gong, Liangliang Zheng, and Sung Wook Kim. *Personal process management: Design and execution for end-users*. Technical Report UNSW-CSE-TR-1018, School of Computer Science and Engineering, the University of New South Wales, Sydney, NSW 2052, Australia, September 2010