SMU
SINGAPORE MANAGEMENT
UNIVERSITY

IS301: Enterprise Integration

Assignment 2
Final Report

by

Tay Wen Bin
Leow Chun Tat
Gabriel Lok
Ong Zhen Guang
Yong Qian Hwee
Adrian Lee

Group 5, Team 3

for

Professor Venky
Instructor ONG Hong Seng

T2 AY 11-12

Introduction

The parking offence process helps Parking Enforcement Officers from the Land Transport Authority (LTA) to determine whether a vehicle that is parked in a parking lot has infringed parking rules. Infringements of parking rules include offences such as the vehicle being parked in a parking lot without a valid season parking and invalid parking coupons.

Our scenario will focus on season parking offences with the following assumptions:

- A valid season parking is one in which the season parking of a vehicle for a particular carpark under a company's charge is within the validity period.
- The carparks in Singapore are either in the charge of the Housing Development Board (HDB) or Urban Redevelopment Authority (URA).
- All parking offences and their corresponding fines can be retrieved from the Traffic Police (TP) Offences and Fines Web Service.
- All Parking Enforcement Officers are under LTA.

Scenario Overview

Scenario Description: The scenario which is the season parking offence process begins when a Parking Enforcement Officer (PEO) checks the season parking of a vehicle in a carpark. The PEO uses his electronic handheld terminal to interface with the Summon Application (SA) and input the vehicle license plate number, carpark id and offence type into their respective fields on the Summon Form. Upon submission of the form, the SA sends a potential summon ticket in the form of an xml formatted string (summon.xml) as a message via the JMS queue request.summon, to the Integration Middleware (IM) which resides in LTA.

The IM receives the potential summon ticket (summon.xml) via the queue, request.summon. The IM validates and parse the potential summon ticket by using its XML schema (summon.xsd). The IM keeps a cached list of carparks (CarparkList.xml). The purpose of caching a list of carparks is so that the IM need not retrieve a new copy from the Carpark Management System for every potential summon ticket it receives. There may be countless of potential summon tickets sent in a day and this will help to prevent the wastage of resources. However, the cached list of carparks has a validity period and after which, it expires and become outdated. This validity period ensures the cached list of carparks is always updated and of the correct version. If the cached list of carparks is outdated, the IM retrieves the correct list of carparks (CarparkList.xml) from the Carpark Management System via FTP. Once the cached list of carparks is updated, the IM validates and parse the cached list with its XML schema (CarparkList.xsd). The IM then uses the carpark id from the potential summon ticket to check against the cached list to find out the company in charge of the carpark.

The IM will invoke the season parking web services (HDB Season Parking Web Service or URA Season Parking Web Service) of the respective company; it submits the vehicle license plate number and carpark id to the season parking web service to check if the vehicle has a valid season parking for the carpark under the company.

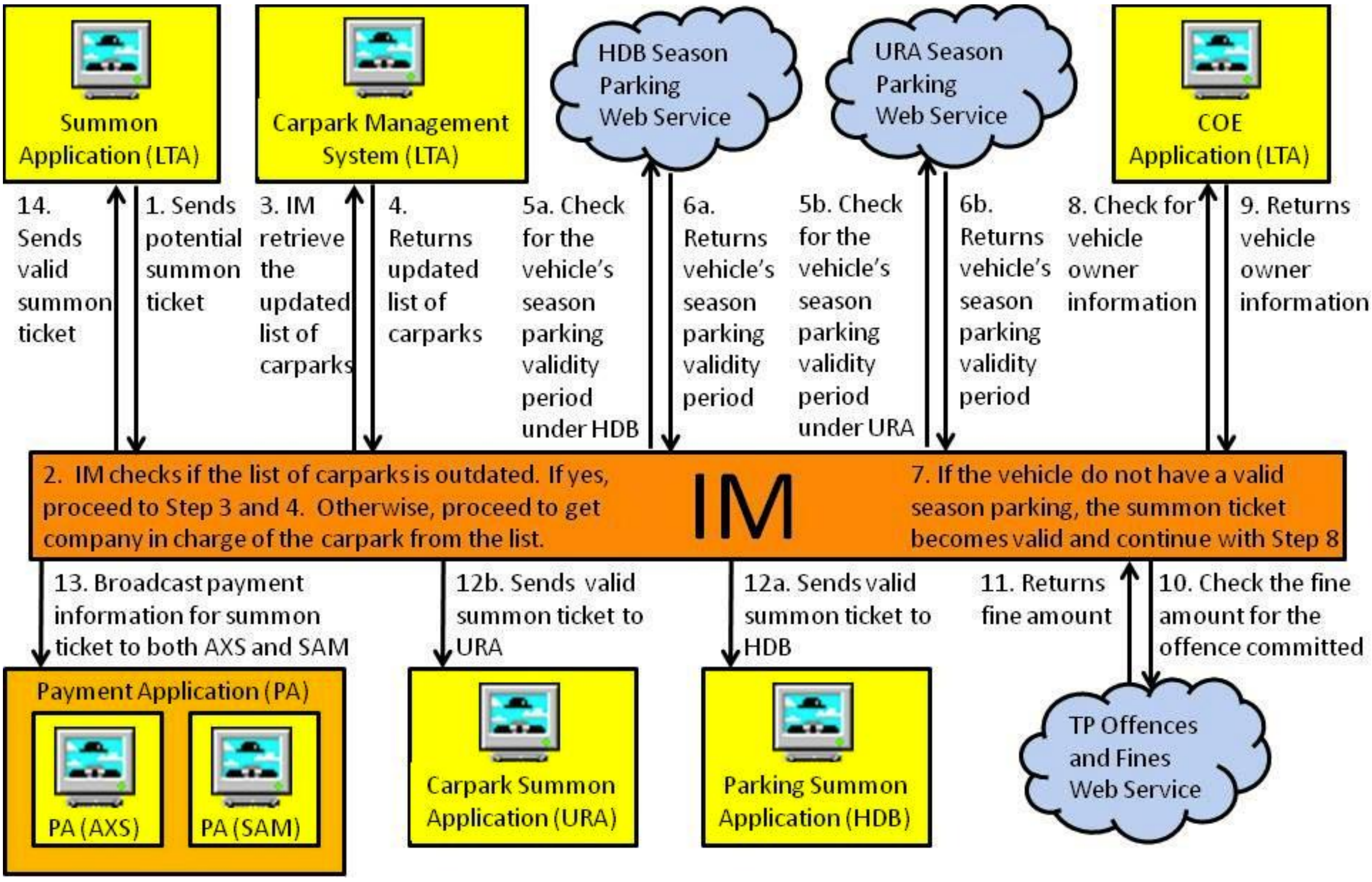
There are two possible outcomes:

1. If the vehicle has a valid season parking for the carpark, no offence has been committed and hence the potential summon ticket becomes an invalid summon ticket. The IM will then render and send the invalid summon ticket details in the form of an xml formatted string (summon.xml) as a message via the JMS queue, reply.summon, to the SA. The PEO who initiated this process can then view the invalid summon ticket from the SA. This marks the end of the season parking offence process.

2. If the vehicle does not have season parking or have an expired season parking for the carpark, it is considered to be an offence and hence the potential summon ticket is now a valid summon ticket. The following steps will be taken:
    a. The IM will send the vehicle license plate number via the queue, request.coeinfo, to the COE Application which resides in LTA. The COE Application will query its database (COEDB) to retrieve information on the vehicle owner. The COE Application then sends the information of the vehicle owner in the form of an xml formatted string (coeinfo.xml) as a message to the IM via the JMS queue, reply.coeinfo.
    b. The IM will invoke and submit the vehicle type and offence type as parameters to the TP Offences and Fines SOAP-based Web Service to obtain the fine amount for this offence.
    c. The IM will render the valid summon ticket into the correct format (hdbsummon.xml for HDB or urasummon.xml for URA) for the company that is in charge of the carpark and hence data transformation has been performed.
    d. The IM will then perform content-based routing and send the valid summon ticket in the correct format (hdbsummon.xml for HDB or urasummon.xml for URA) to the correct company via their respective queues (q.hdbsummon for HDB or q.urasummon for URA). The Parking Summon Application (PSA) resides in HDB and listens to the queue, q.hdbsummon, and the Carpark Summon Application (CSA) resides in URA and listens to the queue, q.urasummon.
    e. The IM will render and send the payment information in the form of an xml formatted string (payment.xml) as a message to the Payment Applications (PA) in AXS and SAM via the topic, t.payment. The PA residing in both AXS and SAM are durable subscribers to the topic, t.payment.

f. The IM will render and send the valid summon ticket (summon.xml) as a message via the JMS queue, reply.summon, to the SA. The PEO who initiated this process can then view the valid summon ticket from the SA.
g. The SA will then store the details of the valid summon ticket into its database (SummonDB)
h. This marks the end of the season parking offence process.

Scenario Diagram:


Summon Application (LTA)
Carpark Management System (LTA)
HDB Season Parking Web Service
URA Season Parking Web Service
COE Application (LTA)
14. Sends valid summon ticket
1. Sends potential summon ticket
3. IM retrieve the updated list of carparks
4. Returns updated list of carparks
5a. Check for the vehicle's season parking validity period under HDB
6a. Returns vehicle's season parking validity period
5b. Check for the vehicle's season parking validity period under URA
6b. Returns vehicle's season parking validity period
8. Check for vehicle owner information
9. Returns vehicle owner information
2. IM checks if the list of carparks is outdated. If yes, proceed to Step 3 and 4. Otherwise, proceed to get company in charge of the carpark from the list.
IM
7. If the vehicle do not have a valid season parking, the summon ticket becomes valid and continue with Step 8
13. Broadcast payment information for summon ticket to both AXS and SAM
12b. Sends valid summon ticket to URA
12a. Sends valid summon ticket to HDB
11. Returns fine amount
10. Check the fine amount for the offence committed
Payment Application (PA)
PA (AXS)
PA (SAM)
Carpark Summon Application (URA)
Parking Summon Application (HDB)
TP Offences and Fines Web Service

Based on the company that is retrieved from the list of carparks with the carpark id in Step 2, the IM will only carry out either 5a or 5b and either 12a or 12b. For example, if the company that was being retrieved in Step 2 is HDB. The IM will invoke the HDB Season Parking Web Service (Step 5a) and sends a valid summon ticket to HDB (Step 12a).

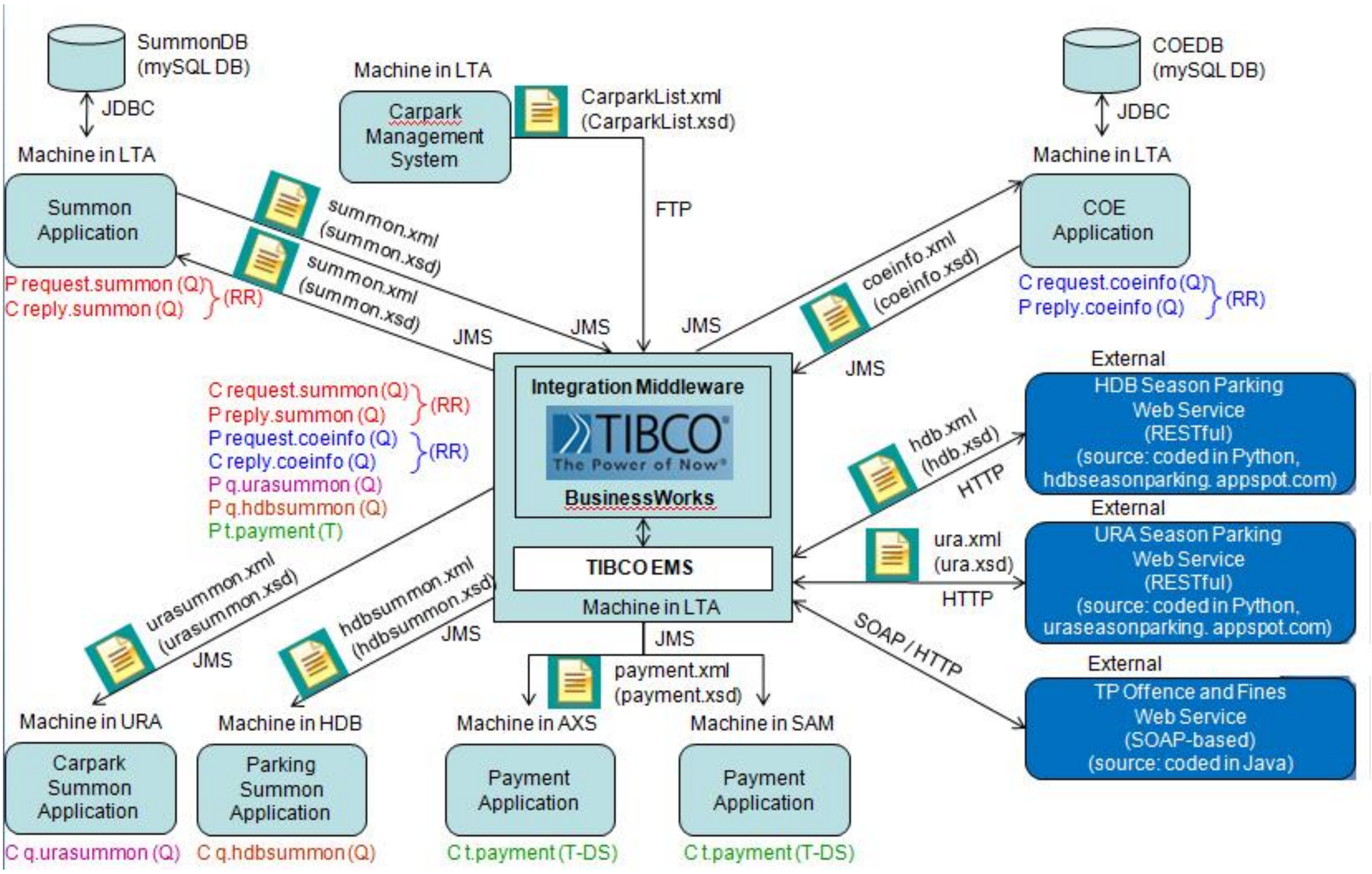
Detailed description of IT Applications and Systems involved

The IT applications and systems involved in the season parking offence process are as follows:

System Name	Description
Summon Application (Location: LTA)	The Summon Application (SA) resides in LTA and it allows the PEO to send a potential summon ticket in the form of an xml formatted string (summon.xml) as a message via a JMS queue, request.summon, to the IM. The SA also listens to the queue, reply.summon, for valid and invalid summon tickets. In addition, it maintains and updates the SummonDB, the database which supports valid summon tickets data.
Parking Summon Application (Location: HDB)	The Parking Summon Application resides in HDB. It receives valid summon tickets which are in the form of xml formatted string (hdbsummon.xml) from the IM that are meant for HDB via the queue, q.hdbsummon.
Carpark Summon Application (Location: URA)	The Carpark Summon Application resides in URA. It receives valid summon tickets which are in the form of xml formatted string (urasummon.xml) from the IM that are meant for URA via the queue, q.urasummon.
COE Application (Location: LTA)	The COE Application resides in LTA and is connected to COEDB, the database which supports COE business data. It receives the vehicle license plate number as a message from the IM via the JMS queue, request.coeinfo. Since it is connected to COEDB, it can query and retrieve information on the vehicle owners of any vehicle that the IM needs from the COEDB. It will then send information of the vehicle owners in the form of an xml formatted string (coeinfo.xml) as a message to the IM via the JMS queue, reply.coeinfo.

Payment Application (Location: AXS and SAM)	The Payment Application resides in both AXS and SAM. For valid summon ticket, it receives the payment information in the form of an xml formatted string (payment.xml) as a message from the IM via the topic, t.payment. In addition, the PA in both AXS and SAM are durable subscribers to the topic, t.payment.
Carpark Management System (Location: LTA)	The Carpark Management System resides in LTA. When the cached list of carparks(CarparkList.xml) in the IM is outdated, the IM will retrieve the updated CarparkList.xml from this system via FTP.

Technical Diagram



JMS Integration

Queue name	Description	Systems involved
q.urasummon(Q)	The IM sends a valid summon ticket in the form of an xml formatted string (urasummon.xml) to the Carpark Summon Application via the JMS queue, q.urasummon.	P – Integration middleware C – Carpark Summon Application
q.hdbsummon(Q)	The IM sends a valid summon ticket in the form of an xml formatted string (hdbsummon.xml) to the Parking Summon Application via the JMS queue, q.hdbsummon.	P – Integration middleware C – Parking Summon Application
request.summon(Q)	The SA sends a potential summon ticket in the form of an xml formatted string (summon.xml) to the IM via the JMS queue, request.summon.	P – Summon Application C – Integration middleware
reply.summon(Q)	After a potential summon ticket is deemed valid, the IM sends a valid summon ticket in the form of an xml formatted string (summon.xml) to the SA via the JMS queue, reply.summon. Similarly, if a potential ticket is deemed invalid, the IM also sends a invalid summon ticket in the form of an xml formatted string (summon.xml) to the SA via the JMS queue, reply.summon.	P – Integration middleware C – Summon Application
request.coeinfo(Q)	The IM sends the vehicle license plate number as a message to the COE Application via the JMS queue,	P – Integration middleware C – COE Application

	request.coeinfo.	
reply.coeinfo(Q)	The COE Application sends information of the vehicle owner in the form of an xml formatted string (coeinfo.xml) to the IM via the JMS queue, reply.coeinfo.	P – COE Application C – Integration middleware

Legend: P – Producer, C – Consumer

We have decided to use queue as the channel for our above JMS interactions because the information needed for the systems flows from point to point. Besides request.coeinfo, all other queues will expect a message in the form of an xml formatted string. A queue channel is sufficient to satisfy our requirement of guaranteed message delivery.

Topic name	Description	Systems involved
t.payment(T)	The IM sends the payment information as a form of an xml formatted string (payment.xml) to the Payment Application (PA) that resides in both AXS and SAM via the topic, t.payment.	P – Integration middleware DS – Payment Application (Resides in AXS) DS – Payment Application (Resides in SAM)

Legend: P – Publisher, DS – Durable Subscriber

We have decided to use topic as the channel for the above JMS interaction. This is because Payment Application (PA) in both AXS and SAM needs the payment information (payment.xml) for the vehicle owner of the valid summon ticket to make the necessary payment. In addition, PA in both AXS and SAM are durable subscribers to the topic, t.payment. This is because if the message is sent when they are offline, they need to receive it when they come back online.

Web Services

1. TP Offences and Fines Web Service (SOAP-based)

Purpose: Provides the fine amount corresponding to the offence type and vehicle type
Input: Vehicle type and offence type
Output: Fine amount for the offence committed based on the vehicle type
Information: This web service is coded in java and hosted on Axis2
Justifications: This is a web service because other parties are also using it. For example, traffic police will also be using this web service for other offences within their charge such as speeding. The IM will invoke this web service; it submits both the vehicle type and the offence type as parameters to this web service. This web service will return the fine amount for the offence committed based on the vehicle type. The advantage of using SOAP is that it is able to handle different transport protocols, including HTTP transport protocol JMS and SMTP.

2. HDB Season Parking Web Service (RESTful)

Purpose: Provides a vehicle’s season parking validity period for a particular carpark registered under HDB
Input: Vehicle license plate number and carpark id
Output: hdb.xml in the form of an xml formatted string
Information: This web service is coded in Python and hosted on Google App Engine
Justifications: This is a web service because other parties are also using it. For example, AXS and SAM will also use this web service for vehicle owners to renew the validity period of the season parking for their vehicle with HDB. This web service is invoked by the IM to check whether a particular vehicle has a valid season parking registered with HDB. The IM invokes this web service as it submits the vehicle license plate number and carpark id to this web service. The web service will return hdb.xml as an xml formatted string to the IM via HTTP. The IM will then parse the xml formatted string. One of the elements in the hdb.xml is the season parking validity period of that particular vehicle. This will enable the IM to determine if the vehicle has a valid season parking period. The advantages of using REST include REST clients being able to access it from any browser and REST application security rules can be setup using http standards.

3. URA Season Parking Web Service (RESTful)

Purpose: Provides a vehicle’s season parking validity period for a particular carpark registered under URA
Input: Vehicle license plate and carpark id
Output: ura.xml in the form of an xml formatted string
Information: This web service is coded in Python and hosted on Google App Engine
Justifications: This is a web service because other parties are also using it. For example, AXS and SAM will also use this web service for vehicle owners to renew the validity period of the season parking for their vehicle with URA. This web service is invoked by the IM to check whether a particular vehicle has a valid season parking registered with URA. The IM invokes this web service as it submits the vehicle license plate and carpark id to this web service. The web service will return ura.xml as an xml formatted string to the IM via HTTP. The IM will then parse the xml formatted string. One of the elements in the ura.xml is the season parking validity period of that particular vehicle. This will enable the IM to determine if the vehicle has a valid season parking period. The advantages of using REST include REST clients being able to access it from any browser and REST application security rules can be setup using http standards.

XML Documents

XML name	Usage
CarparkList.xml	To record the details of the carpark id and the corresponding company (HDB or URA) which manages that carpark.
coeinfo.xml	To record the details of the vehicle owner whose license plate number has been enquired in a valid summon
hdb.xml	To record the details of the vehicle's season parking period in which its license plate number has been determined to be registered under HDB
hdbsummon.xml	To record the details of the vehicle owner, his offence type and the fine amount in which his vehicle license plate number has been enquired in a valid summon. The carpark id in the valid summon is determined to be registered under HDB.
payment.xml	To record details of the vehicle's owner fine amount for the season parking offence he has committed. His license plate number has been involved in a valid summon
summon.xml	To record the details of the vehicle license plate number, the carpark id and the offence type of the potential summon.
ura.xml	To record the details of the vehicle's season parking period in which its license plate number has been determined to be registered under URA
urasummon.xml	To record the details of the vehicle owner, his offence type and the fine amount in which his vehicle license plate number has been enquired in a valid summon. The carpark id in the valid summon is determined to be registered under URA.

Integration Middleware – Requirements

Data Transformation: A summon ticket is represented using different structures in different systems; the SA uses summon.xml while the Parking Summon Application in HDB and the Carpark Summon Application in URA uses hdbsummon.xml and urasummon.xml respectively. Data transformation is typically performed by the IM in which the data is being changed from one structure to another structure. The SA sends a potential summon ticket in the form of a xml formatted string(summon.xml) as a JMS message to the IM. The IM will parse the summon.xml and continue with a series of activities inside the IM to determine whether a potential summon ticket is a valid summon ticket. In the event that the potential summon ticket becomes a valid summon ticket, the IM will then send it to either HDB or URA. However, HDB and URA are using different xml data structures for their valid summon tickets. If the valid summon ticket is meant for HDB or URA, the IM will render the valid summon ticket to either hdbsummon.xml or urasummon.xml respectively according to their xml data structure; data transformation is hence performed.

Content-based Routing: The SA sends a potential summon ticket with its details in an xml formatted string (summon.xml) as a JMS message to the IM. The summon.xml can be issued for a vehicle that is parked either in a HDB or URA carpark. The IM has a cached list of carparks (CarparkList.xml). The CarparkList.xml contains all the carparks under both HDB and URA. The carpark element in CarparkList.xml has two child elements, carpark id and the company which is in charge of this carpark. The IM will then use the carpark id which is obtained from the potential summon ticket to check against the list of carpark ids in the CarparkList.xml to find out which company the carpark belongs to. After the IM determines that the potential summon ticket is a valid summon ticket through a series of activities, the IM will render the summon ticket to either hdbsummon.xml or urasummon.xml respectively according to their xml data structure, performing data transformation. With the company name, the IM will then be able to route the summon ticket to the correct company using JMS; content-based routing based on the company name is hence performed.

Integration Middleware – Implementation beyond the Lab

Call Process Activity: We placed the common series of activities in different process definitions such as "Offences Process", "Payment Process" and "Reply Process". The Offences Process is used to obtain information on the owner of a vehicle that does not have a valid season parking as well as the fine amount that corresponds to the offence committed based on vehicle type. The Payment Process is used to send information via topic to the Payment Applications in AXS and SAM. The Reply Process is used to send a valid/invalid summon ticket to the SA via queue, reply.summon so that the PEO who submitted the potential summon ticket can view it from the SA. The Main Process will then use the "Call Process Activity" to call and execute the other three process definitions, Offences Process, Payment Process and Reply Process in order to complete the Season Parking Offence process.

Group & Iterate Loop: We grouped a series of activities and used the group action, "Iterate Loop", so that the IM will loop through the cached list of carparks (CarparkList.xml) to find the corresponding company for the carpark id that is submitted in the potential summon ticket (summon.xml). This is necessary as it will allow the IM to invoke the Season Parking Web Service of the correct company and in the event that the potential summon ticket becomes valid, it will enable the IM to route the valid summon ticket (hdbsummon.xml for HDB or urasummon.xml for URA) to the correct company via their queues.

Caching and FTP Get: The IM caches a copy of the list of carparks (CarparkList.xml). Caching a copy of the list of carparks means that the IM need not retrieve a new copy from the Carpark Management System for every potential summon ticket it receives. This helps to prevent wastage of resources as there may be countless of potential summon tickets sent in a day. When the cached list of carparks (CarparkList.xml) is outdated, we used the FTP Get activity inside the IM to issue an “FTP get” to the Carpark Management System to retrieve the updated CarparkList.xml.

XPath’s String Functions: We concatenated the valid-till date and time from the cached list of carparks (CarparkList.xml) by using “substring-before(<< string >>, << end-string >>)” and “substring-after(<< string >>, << starting-string >>)” for the date and time respectively. This concatenated “dateTime” string is the expiry date of the CarparkList.xml and will be used to ensure that the cached CarparkList.xml is of the correct version.

XPath’s Date/Time Functions: The “current-date()” returns the current local date as a date string. The “tib:compare-date(<< date1 >>, << date2 >>)” is used to compare today’s date with the expiry date of the season parking. Based on the comparison of dates, the IM will then be able to determine if the vehicle has a valid season parking. The “current-dateTime()” returns the current local time as a dateTime string. The “tib:compare-dateTime(<< dateTime1 >>, << dateTime2 >>)” is used to compare the current date and time with the expiry date of the cached list of carparks (CarparkList.xml). Based on the comparison of dates, the IM will then be able to determine if the CarparkList.xml is outdated. This is to ensure that the cached CarparkList.xml is of the correct version before using it.

Others – Implementation beyond the lab

RESTful Web Services: Our HDB and URA Season Parking Web Services are RESTful web services. Our IM is able to invoke these web services to check whether the vehicle has a valid season parking for the carpark under the corresponding company. This is part of our group’s initiative to explore RESTful web service.

Web Services coded in Python: In addition, both HDB and URA Season Parking Web Services are coded in Python. This demonstrates the role of XML documents in integration as they are loosely coupled. Despite these web services being coded in another language besides Java, the IM is still able to invoke them and retrieve data in XML documents. This is because XML documents are language independent.

Web Services hosted on Google App Engine (GAE): Furthermore, both HDB and URA Season Parking Web Services are also hosted on GAE. In reality, these two web services will not be hosted on GAE due to security concerns. However, we have hosted these two web services on GAE to simulate integration across different platforms. The IM invoke these web services and retrieve data in XML documents. This highlights the role of XML documents in integration as well. The integration across different platforms is possible because XML documents are platform independent.