INTEGRATION WHITE PAPER: Wynne Systems RentalResult

June 2017

2601 Main Street, Irvine, CA 92614
Sales: +1 949 224 6300, ext. 2
Service Desk: +1 866 901 9284

Table of CONTENTS

Integration Tools

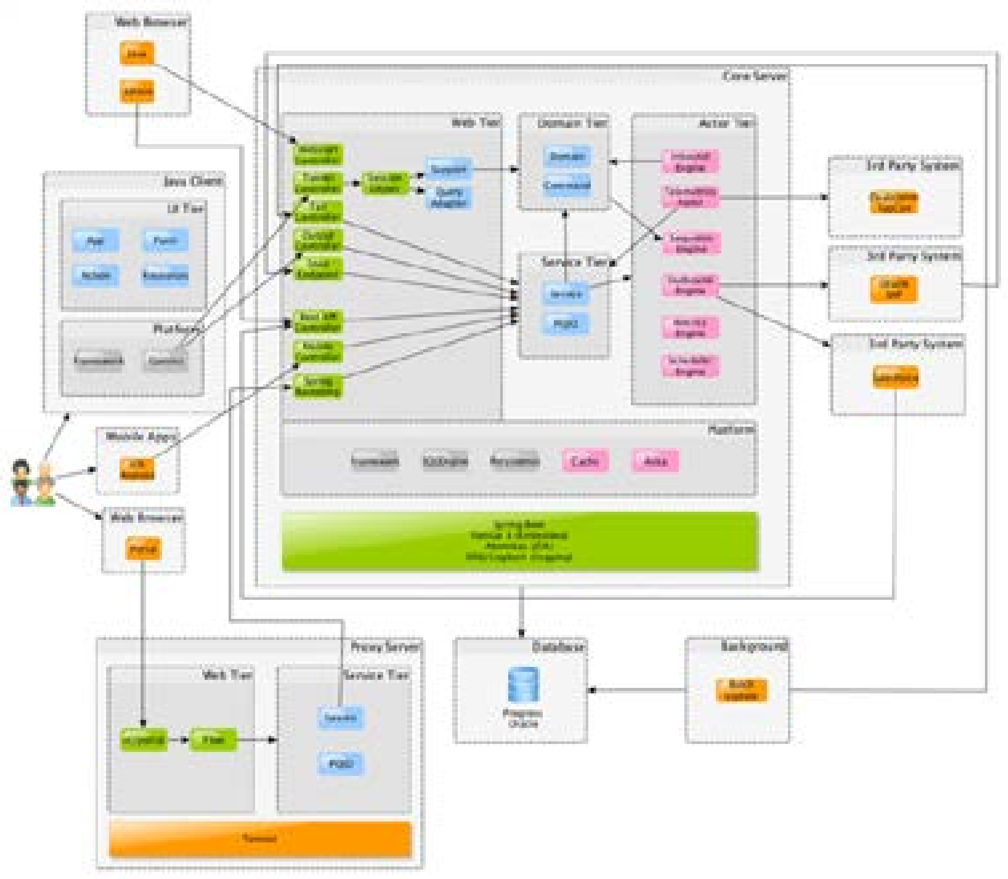
Introduction

The Wynne RentalResult suite has a full web services platform using SOAP or RESTful web services to manage data movement between systems. For those systems where real-time communication is not appropriate or possible we can also interface using flat file or scheduled transactions. This document concentrates on ERP and point based Integration utilizing SOAP or RESTful web services.

A web services exchange example is included at the end of the document in the telematics section.

Integration Architecture

The Wynne RentalResult system is a modern N-tier architecture based on the Java (JEE)/Spring Framework platform and is composed of the following layers as shown below.



Integration Tools

The following core components make the integration platform:

Credit Card: the Wynne RentalResult server can take credit card payments as part of some business processes via a credit card payment gateway interface plugin mechanism. Currently Paymetrics and Cybersource are supported but new plugins can be written for other payment gateways.

Outbound Interfaces: Trigger points in the business logic can cause XML messages to be sent to the ESB for routing to 3rd party systems (accounts, invoices, etc.). The transport mechanism can differ depending on the customer integration requirements. For instance we have integrated with Oracle and SAP using SOAP over HTTP and with IBM WebSphere MQ using SOAP over JMS.

Telemetric: the Wynne RentalResult server can periodically connect to telemetric services such as Qualcomm or Topcon to collect asset sensor/geolocation information.

Tax Adapter: Wynne RentalResult can be configured to connect to a Vertex tax web service.

Spring Remoting: for in-house Java applications that need to run on separate servers (such as the Wynne RentalResult e-commerce server), the spring remoting protocol can be used for Java to Java fast RPC calls.

REST API endpoints: the Wynne RentalResult system provides a REST API for native mobile applications (Android, iOS), native Java or non-Java applications (.NET WinForms apps) and web applications to access resources and update data in Wynne RentalResult. This makes it possible for Wynne RentalResult to coexist in a complex web environment where data needs to be collected (mashed) from various systems but presented in a unified user UI web experience. This makes the architecture completely open to programming by following the best REST design and principles.

File Batch Import: sometimes 3rd party systems need to import into Wynne RentalResult large amounts of data (assets/accounts) at periodic intervals. CSV type files can be scanned in shared network directories for that purpose.

Inbound Interfaces: traditional SOAP type web services are available for enterprise to enterprise integration. This is used for importing data in real-time into Wynne RentalResult. In a lot of our integration projects, a 3rd party system like Oracle or SAP is set up for bidirectional integration with Wynne RentalResult. For instance Oracle would export A/R accounts to Wynne RentalResult by calling the ImportAccount web service and Wynne RentalResult would call an Oracle web service whenever an account is updated in Wynne RentalResult (via the outbound interface component and the ESB transform/routing technology).

Security Flow for Integration Architecture

The RentalResult architecture utilizes the Spring OAuth2 extension library to provide OAuth2 authentication to the RentalResult platform. We currently have the 2 following flows supported:

Password flow
The users RentalResult credentials are passed to the /OAuth/token endpoint using the password grant type. A bearer token is passed back to the client application and each subsequent calls must contain the bearer token inside the authorization HTTP header parameter. This flow is used by the iOS/Android mobile applications, where the username/password are entered by the user using a login screen. All communications to the backend server are conducted over HTTPS.

JWT Bearer token flow
This is a custom flow used at RentalResult to secure our REST and SOAP API layers. The client application programmers are given a key by the RentalResult support team. The key must be used to compute a JWT assertion that is then passed to the /OAuth/endpoint token. Just like the password flow, in the JWT bearer flow, a bearer token is sent back to the client application and must be passed as part of subsequent REST/SOAP calls using the authorization HTTP header parameter

High Level Examples of Existing ERP Integrations

Oracle & Wynne RentalResults

The depth of your integration between Oracle and RentalResult is entirely up to you. Existing implementations range from simple master data (project & customer) and GL postings, to real time complex integration at every level. This section aims to lay out 2 scenarios, currently used by different Oracle customers who use Wynne RentalResult to run their equipment management businesses.

Scenario 1: Full Scale Integration to Oracle EBS
Where the equipment business is tightly bound into the corporate entity, or where the equipment business is the corporate entity we normally expect a full scale integration between the systems. Master data is largely managed in Oracle, vendor, customer, project, asset and financial data all pass between the systems in real time and user roles encompass both applications.

General Ledger	The Oracle GL is the only system of record, data is fed to the GL through the subsidiary ledgers.

AR & Billing	Invoices and charge documents relating to equipment movement/ usage, rentals or operation equipment timesheets are raised in RentalResult and sent through to Oracle AR. Credit is managed though Oracle unless you specifically want to manage a separate equipment credit limit through RentalResult.
AP & Purchasing	If you are renting equipment in from external vendors then you need to manage rental purchase orders. RentalResult can manage this directly from within the equipment management process and make the Oracle AP matching process as streamline as possible. We create and approve the PO in RentalResult and replicate into Oracle. We then create goods receipt documents in Oracle for each billable period of time as it is reached so that you can apply multiple date based purchase invoices to the PO. All purchasing assets destined for the asset register is handled in Oracle, with Oracle as the financial system of record for your assets.
Projects, Internal Charges and Accruals	Oracle is the system of record for all project data. All activity in RentalResult can be linked to project information and any data sent back to Oracle will include project details. If you need to provide job sites or customers with weekly or monthly project accruals by task codes then we can generate the relevant accruals for you and interface to Oracle projects.
Financial Asset Management	In this scenario we would expect all financial asset management to be done in Oracle unless you have a different preference. Oracle holds the master data.
Equipment Mobilization and Physical Asset Management	All transactional events which affect location, status or usage of the asset are managed through RentalResult. This means that the physical asset it managed in RentalResult. Logistics, movements on and off job sites, rental deliveries and confirmations, and the off-rent or de-mobilization process are all managed through RentalResult. This includes all mobilization and physical management of externally rental assets and may include consumables.
Maintenance and Servicing	If the only maintenance that you need to manage relates to the equipment that you are managing through RentalResult we would recommend that you utilize RentalResult maintenance and service applications. In which case anything that affects the value of an asset – i.e.: if you do any refurbishing work, or anything that add tot eh asset costs then the data will be sent though to Oracle. If you are already using Oracle's maintenance and servicing applications then instead we will use transactional and status change events to trigger activities in Oracle. Asset status can be used to all or disallow behavior in RentalResult. For example; until a service is confirmed as being performed in Oracle the user will not be allow to dispatch the asset in RentalResult.

Scenario 2: Limited Integration to Oracle EBS
In some instances we have customers who are looking for an integration which is limited to specific key function and data. This tends to be where the equipment management business is acting as a sub-contractor to the main business and there is a degree of distance required between the two businesses. In this instance we often find the groups of people using Oracle and Wynne RentalResult are almost entirely distinct, there is virtually no overlap of systems from an end user point of view.

General Ledger	The Oracle GL is the only system of record, data is fed to the GL through the subsidiary ledgers and from RentalResult AR and asset register.
AR & Billing	Invoices and charge documents relating to equipment movement/ usage, rentals or operation equipment timesheets are raised in RentalResult and the credit collection process is managed entirely in RentalResult. Credit is managed in RentalResult.
AP & Purchasing	All purchasing of equipment, rental purchasing and consumables for maintenance and servicing are managed within RentalResult. Purchase orders are approved, received and matched within RentalResult. Matched invoices are sent through to Oracle AP for payment.
Equipment Mobilization and Physical Asset Management	All transactional events which affect location, status or usage of the asset are managed through RentalResult. This means that the physical asset it managed in RentalResult. Logistics, movements on and off job sites, rental deliveries and confirmations, and the off-rent or de-mobilization process are all managed through RentalResult. This includes all mobilization and physical management of externally rental assets and may include consumables.
Maitnenance and Servicing	All ad-hoc and scheduled maintenance is carried out using RentalResult maintenance and servicing applications. Anything that affects the value of an asset – i.e.: if you do any refurbishing work, or anything that adds to asset costs, costs of parts and labor on schedule maintenance will be sent through to Oracle as a journal.

Example of Oracle Integration
In October 2011, customer X (available as a reference) introduced a new financial software solution across all their businesses. This solution involved the integration of Wynne RentalResult, their rental fleet operational management software, with a financial solution provided by Oracle which includes R12 and OLFM. The integration between RentalResult and the Oracle solution allowed for information crucial to handling the financials to flow automatically from RentalResult to Oracle and any financial information that affects operations to flow back from Oracle.

The integration was achieved with the use of web services transferring industry standard SOAP based messages over HTTP. The solution included a number of both inbound services allowing for data to be pushed out of Oracle to RentalResult and outbound services allowing for data to be pushed to Oracle based on the appropriate trigger being activated in RentalResult.

The services within RentalResult connected to a number of services created within Oracle's middleware SOA suite. This allowed Oracle to wrap up a number of their API's and open interfaces into SOAP based web services as well as perform both inbound and outbound message transformation. The table below summaries the interfaces between RentalResult and SOA giving a summary of the functionality provided. This along with the interface diagram gives an overview of the interfaces available and data being transferred in the Customer X solution.

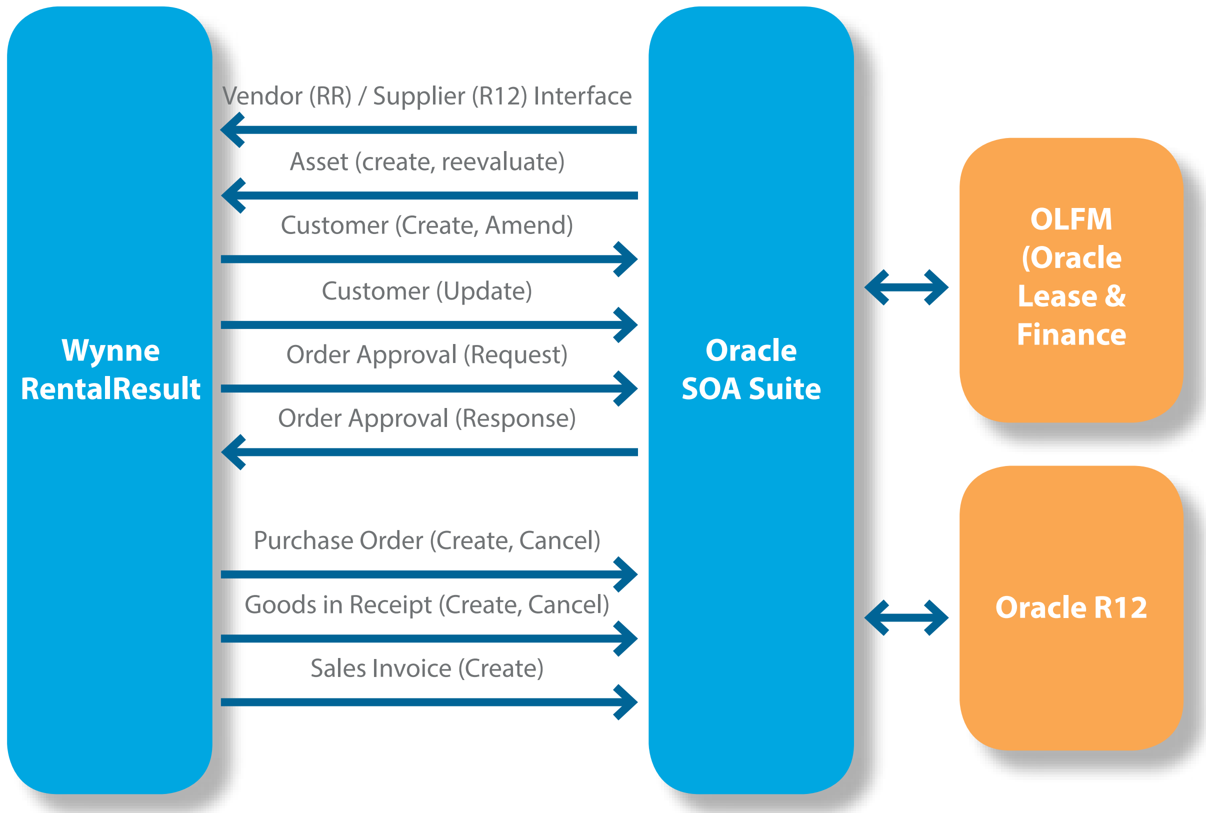
Interface Name	Description	Interface Direction
Vendor	Creates and updates vendor accounts in RentalResult based on supplier data exported from Oracle	Oracle to RentalResult
Contract Credit Approval	Initiates order approval process in Oracle and allows for decision to be updated against contract in RentalResult	Both
Customer Credit	Updates credit status on customer accounts in RentalResult	Oracle to RentalResult
Customer	Creates an account in Oracle based on account data in RentalResult	RentalResult to Oracle
Asset	Creates an asset in RentalResult and allows for revaluation of the asset based on external depreciation	Oracle to RentalResult
Purchase Order	Creates purchase order in Oracle based on PO data in RentalResult	RentalResult to Oracle
Goods in Receipt	Creates assets in Oracle on receipt of assets into RentalResult	RentalResult to Oracle
Sales Invoice	Exports sale invoice to Oracle and retires assets at sale	RentalResult to Oracle

• As you'll see from the diagram below, it was decided that where purchasing related directly to a rental or sales order it was triggered from RentalResult and then used to electronically create a PO in Oracle, the receipt was then performed in RentalResult and again electronically pushed to Oracle.

• Where purchasing was driven more by the business – i.e.: replenishment of the rental fleet, of parts etc., the PO was initiated in Oracle and rental assets are received in Oracle and then sent through to RentalResult.

- This concept is something that we have also used elsewhere particularly around sub rentals where Oracle's difficulty with matching multiple invoices to a single GRN means that the slickest solution is to manage the whole of the sub rental purchase and receipt in Wynne RentalResult, interface the documents electronically to the Oracle AP, and then auto-send a new goods receipt and PO line for each sub-rented asset if it stayed on rent beyond the initial period to allow Oracle AP matching to occur seamlessly.

RentalResult/Oracle Integration at Customer X

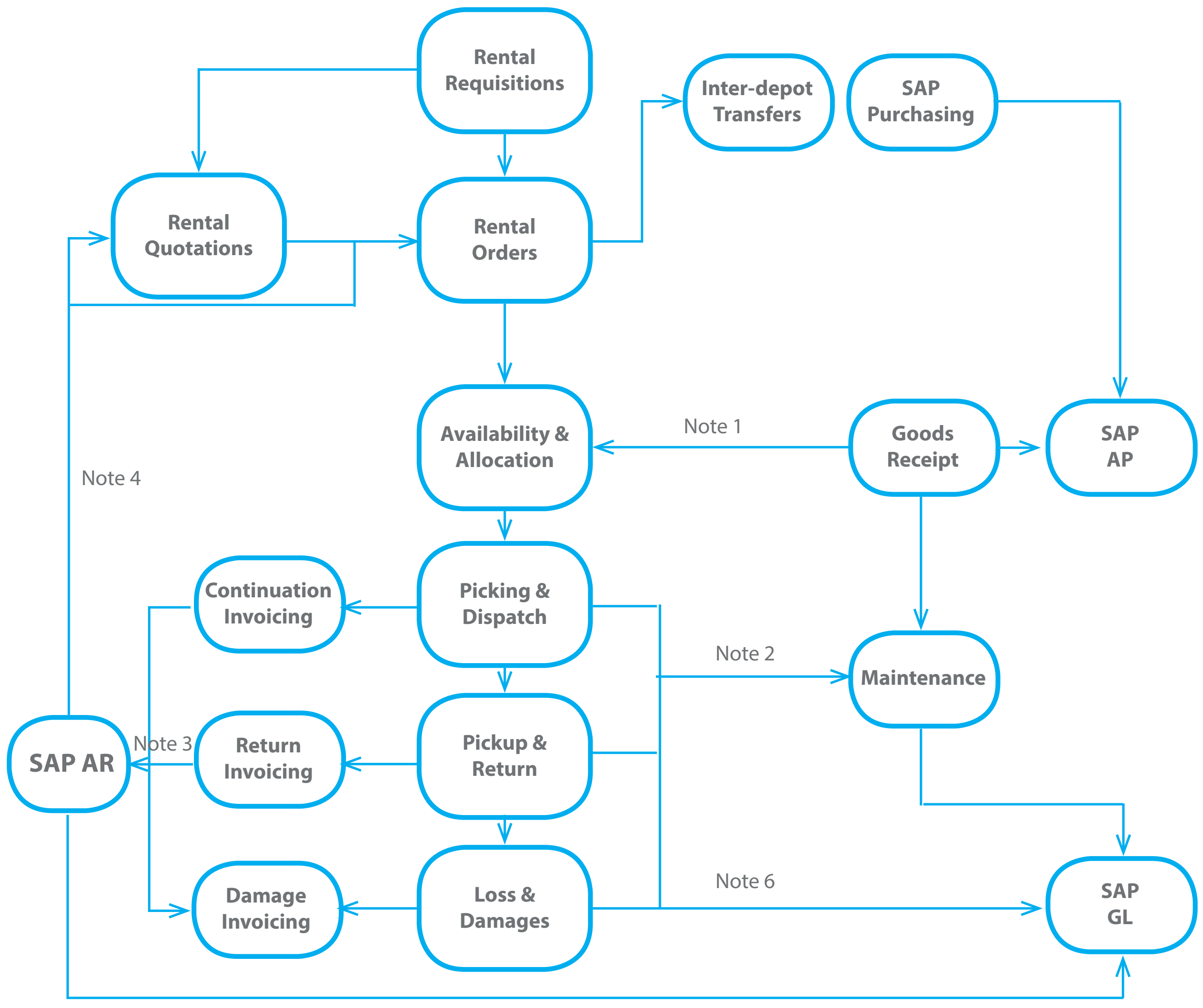

Wynne RentalResult
Vendor (RR) / Supplier (R12) Interface
Asset (create, reevaluate)
Customer (Create, Amend)
Customer (Update)
Order Approval (Request)
Order Approval (Response)
Purchase Order (Create, Cancel)
Goods in Receipt (Create, Cancel)
Sales Invoice (Create)
Oracle SOA Suite
OLFM (Oracle Lease & Finance
Oracle R12

SAP & Wynne RentalResult

The diagram and notes below provide an example of the type of integration we have with SAP for a large rental company in the USA (available as a reference). We also support some customers still feeding data to SAP via old-style BAPI interfaces and with much more simplified structures. For example AR invoices and asset values only. As in all cases, it is easy to tailor the Wynne RentalResult interfaces to fit your specific business requirements.

Scenario 1: Full Scale Integration to Oracle EBS
Where the equipment business is tightly bound into the corporate entity, or where the equipment business is the corporate entity we normally expect a full scale integration between the systems. Master data is largely managed in Oracle, vendor, customer, project, asset and financial data all pass between the systems in real time and user roles encompass both applications.

Example of SAP Integration



Note 1: Assets are bought and received into the asset register in SAP. The asset master resides in SAP. At a point in time the assets are marked as being part of the rental fleet in SAP and this triggers an interface into RentalResult to make the asset visible to the rental system. At this point the rental system takes physical ownership of the asset, and any status changes against the asset are managed in the rental application until such time as it is sold, lost or retired from the rental fleet.

Note 2: Maintenance activity is normally carried out in the SAP system, but can be done within RentalResult if necessary. You will capture service units such as clock hours within the rental process and this information is sent to the SAP system to assist with management of maintenance activities.

Note 3: All rental invoices and billing data is captured and created in RentalResult and the invoices are then sent straight through to the SAP AR. All Cash collection, etc. is performed in SAP.

Note 4: The customer master resides in SAP and is interfaced directly through to the RentalResult application in real time, so that credit checks etc. can be performed directly from Quotation and Rental Contract entry in RentalResult. In the simple model a new customer needs to be set up in SAP before it can be used in RentalResult. You can also use a two interface which would allow a customer to be created in RentalResult and then sent through to SAP.

Note 5: We do not update the GL directly from the billing process, SAP's interfaces are structured so that when you load billing documents into the AR, the AR itself makes its own postings. We can supply the invoice data down to the lowest level of detail (asset number etc.) so that your postings can reflect your business requirements.

Note 6: If any data is captured at an asset level that needs to be reflected in the SAP GL or Asset Register this is sent as journal data. This may include damage charges, loss information or other values which are captured naturally as part of the rental process.

High Level CRM Integrations

Wynne RentalResult with CRM Applications

Scenario 1: Integration of Customers, Prospects & Contacts

This level of integration provides basic data flow between the Wynne RentalResult platform and the CRM system of your choice, existing customers utilize Salesforce, Zoho and MS Dyamics.

If your CRM system is used for tracking customer interactions, mailshots, etc. then this type of integration will serve your needs.

Integration between CRM & Wynne RentalResult

Lead/Customer/Contact Data from CRM to RentalResult

• Create new lead in RentalResult (link to quote)
• Amend lead in RentalResult
• Amend customer details
• Request creation of customer from lead
• Create contract in RentalResult
• Amend contract in RentalResult

Lead/Customer/Contact from CRM to RentalResult

• Request list of leads from RentalResult
• Request specifric lead from Rental
• Amend customer details
• Request creation of customer from lead
• Create contract in RentalResult
• Amend contract in RentalResult

Scenario 2: Integration of Detailed Transactional Information

This takes CRM interaction to a much greater level. It is only necessary if you want to be able to pass detailed quote information between the applications. This means that effectively you need to be able to replicate product, pricing and availability information into your CRM application.
This is not recommended for all customers, you will need considerable skills on the Salesforce integration side.

Integration between CRM & Wynne RentalResult

Quote Related Data from CRM to RentalResult

- Send new quote to RentalResult
- Send modified quote to RentalResult
- Send converted quote command
- Send lost opportunity command

Quote Related Daya from CRM to RentalResult

- Request list of quotes for lead or prospect
- Request quote details
- Request individual or list of products
- Request availability information for products
- Request pricing and rate information for products

Wynne RentalResult with Telematics

Telematics Background Poller

The RentalResult application can be configured to poll a given telematics provider for up-to-date equipment data. The mechanism is generic so that any telematics provider can be integrated. We have put in place an internal plugin mechanism and currently have implemented plugins for Qualcomm and Tierra Topcon. A plugin is simply a facade against the telematics vendor API (REST or SOAP). Most of our integration code is in place and if another vendor or solution needs supporting, it is simply a matter of identifying the API hooks to call and to data map from the vendor API messages to our internal telematics objects.

A typical flow for a plugin is the following, for each polling cycle:

1) Authenticate to the vendor REST API
   a. This could be simple basic authentication for each web service request to the vendor for the current polling cycle
   b. Pass credentials and get an access token which is kept in memory and subsequently passed to each further web service request to the vendor for the current polling cycle.

2) Get equipment latest data by calling the relevant vendor API, and for each retrieved equipment data payload:

a. Lookup the RentalResult asset from the equipment ID
b. Create a new location record within RentalResult for the asset
c. Create a new engine on/off record within RentalResult for the asset
d. Update the total operating hours of the asset within RentalResult
e. For each registered sensor for the asset, update the corresponding sensor record within RentalResult (for instance fuel used in the last 24hrs, Proximity, etc.)

The Telemetric plugin also provides a facility to ping an existing piece of equipment ID to retrieve up to date information for a specific asset. The mechanism used is the same as the telemetric poller, except that it is for just a given equipment, as opposed to a list of them.

For reference, following is an example Request/Response set of payloads for our Topcon plugin when retrieving a specific equipment data:

Real-time Asset Ping

```
<soapenv:Envelope xmlns:soapenv=”http://schemas.xmlsoap.org/soap/
envelope/” xmlns:tier=”http://www.tierraservice.com/”>
   <soapenv:Header/>
   <soapenv:Body>
      <tier:GetFleetData>
         <tier:equipmentUidList>
            <tier:unsignedLong>2923</tier:unsignedLong>
         </tier:equipmentUidList>
      </tier:GetFleetData>
   </soapenv:Body>
</soapenv:Envelope>

Response:
<soap:Envelope xmlns:soap=”http://schemas.xmlsoap.org/soap/
envelope/” xmlns:xsi=”http://www.w3.org/2001/XMLSchema-instance”
xmlns:xsd=”http://www.w3.org/2001/XMLSchema”>
   <soap:Body>
      <GetFleetDataResponse xmlns=”http://www.tierraservice.
com/”>
         <GetFleetDataResult>
<![CDATA[<?xml version=”1.0”?><
            <Fleet version=”1.2” snapshotTime=”2012-04-
16T19:32:12Z” xmlns=”http://schemas.aemp.org/fleet”>
               <Equipment>
                  <EquipmentHeader>
                     <EquipmentUID>2923</EquipmentUID>
                     <Make>ZOOM BOOM</Make>
                     <Model>ZBM20044T3</Model>
                     <EquipmentID>ZB-08103</EquipmentID>
                     <SerialNumber>B20044T307580F</SerialNumber>
                  </EquipmentHeader>
                  <Location datetime=”2012-04-14T17:10:05Z”>
                     <Latitude>34.0711</Latitude>
                     <Longitude>-81.0230</Longitude>
```

```
        </Altitude>
                     <AltitudeUnits>meters</AltitudeUnits>
                     <Address>
                        <Street>Fairfield Rd</Street>
                        <PostalCode>29203</PostalCode>
                        <City>Columbia</City>
                        <AdministrativeArea>SC</
AdministrativeArea>
                        <Country>USA</Country>
                     </Address>
                  </Location>
                  <CumulativeOperatingHours datetime="2012-04-
11T17:09:24Z">
                     <Hour>PT22H46M12S</Hour>
                  </CumulativeOperatingHours>
                  <FuelUsedLast24 datetime="2012-04-
16T17:32:08Z">
                     <FuelUnits>gallon</FuelUnits>
                     <FuelConsumed>0</FuelConsumed>
                  </FuelUsedLast24>
                  <Distance datetime="2012-04-14T17:10:05Z">
                     <OdometerUnits>mile</OdometerUnits>
                     <Odometer>78</Odometer>
                     <ResetDateTime>2011-03-02T18:20:26Z</
ResetDateTime>
                  </Distance>
               </Equipment>
            </Fleet>
]]>
         </GetFleetDataResult>
      </GetFleetDataResponse>
   </soap:Body>
</soap:Envelope>
```

Asset Alerts Web Service

In addition to the poller, the RentalResult SOAP API provides an endpoint for updating asset alerts within RentalResult. This is so that the telemetrics provider can send real-time notifications to RentalResult when equipment alerts occurred in the field.

The WSDL can be accessed at the RentalResult server via HTTP as follows:

GET /ws/soap/assetAlert.wsdl

Below is an example Request payload that RentalResult will accept (subject to the usual authentication to the RentalResult API via oauth bearer tokens):

```
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/
soap/envelope/" xmlns:ass="http://www.RentalResult.com/schema/
assetalert">
   <soapenv:Header/>
```

```
<soapenv:Body>
      <ass:ImportAssetAlertRequest>
         <ass:AssetAlerts>
            <ass:AssetAlert datetime=”2012-04-15T07:49:36Z”>
              <ass:EquipmentId>JLT-07502</ass:EquipmentId>
              <ass:Type>Curfew</ass:Type>
              <ass:Status>B</ass:Status>
              <ass:Add¬itionalInfo>The unit is entering fence :
Turin</ass:AdditionalInfo>
            </ass:AssetAlert>
            <ass:AssetAlert datetime=”2012-04-16T13:40:36Z”>
              <ass:EquipmentId>GN-1101</ass:EquipmentId>
              <ass:Type>Fence</ass:Type>
               <ass:Status/>
               <ass:AdditionalInfo>PTL Test Data GN-1101</
ass:AdditionalInfo>
            </ass:AssetAlert>
         </ass:AssetAlerts>
      </ass:ImportAssetAlertRequest>
   </soapenv:Body>
</soapenv:Envelope>
```