Whitepaper: extended warranties – a blockchain solution

Team: NChain

Brief Summary

NChained have produced a blockchain solution to handle Extended Warranties. We see there being benefits for all actors, namely the customer, retailer, service agent and Zurich Insurance itself.

The benefits are found by having a centralised system where information is available to everyone, immediately and securely. Human decisions are replaced by smart contracts, meaning a drastically reduced cost to Zurich. The increased simplicity encourages retailers to sell more policies, whilst a pricing model that learns from the claims experience and updates itself brings an improved loss ratio.

This paper examines these benefits in more detail, and gives the technical background behind the solution.

Extended Warranties

# The Problem

Zurich offers an Extended Warranty product. These policies are purchased to complement the Manufacturer’s Warranty (typically two years) on an appliance or similar good, and offer cover against breakdowns for a further one to three years after the Manufacturer’s Warranty expires. They are usually sold by the retailer at the time of the original purchase, but can be purchased at a later date provided the Manufacturer’s Warranty is still in force.

The nature of the business is a high volume, low price product. The low price means minimal-touch administration is needed.

The problem is that there are many retailers, each of which will have records of what they sell, as will Zurich Insurance. A lot of time is spent trying to reconcile this data. The problem is compounded when a claim is made, and the service agent (repairer) has to check what is covered before fixing the product.

A solution is required which can:

* keep an irrefutable record of exactly what has been sold and to whom, accessible by all parties
* price the policy correctly, according to the product’s claim history, length of policy, commission rates and other underwriting factors
* record commissions paid to the sales agent
* Allow a customer to easily find details of their policy, even years after they purchased it and lost the original paperwork (frequently a piece of paper stapled to the receipt)
  + show the version of the policy that the customer purchased (policies change over the years, we must be able to recreate the policy that was in force at the time the customer purchased it)
* allow a customer to log a claim, and for these details to be visible to everyone
* help the service agent handle the claim

In short, make this line of business very straightforward and inexpensive to administer.

To facilitate minimal human touch, the system should apply rules (smart contracts) to handle the following:

* price calculations, which can change by product and retailer, and by the length of time which has passed after the initial purchase (the price will increase the further we are into the manufacturer’s warranty)
* check coverage, once the customer and / or service agent provide the details of the claim
* calculate the likely cost of a repair and decide whether to repair or replace accordingly

In addition, the system should apply rules to help detect fraud or other problems. These will include flagging cases:

* where a service agent is frequently charging more than other service agents for similar repairs
* where a customer has a higher than expected volume of claims
* where a particular brand of product has a high breakdown rate

The reporting requirements vary according to which party is running the report, but the following reports will be required:

* Certificate production (a certificate will be emailed to the customer when the product is purchased)
* Identifying customers who didn’t purchase the Extended Warranty at the time of appliance purchase, but who are still in time to make a purchase. We should also report the conversion rate of such marketing approaches
* A complete sales history, by date, split by retailer as appropriate
* Analysis around:
  + Products - best-selling, most reliable and least-reliable
  + retailers – which are selling the most / fewest policies, and why might this be
  + Service agents – who carries out the most repairs, who recommends more fixes than replacements, who charges most per repair

# The Solution – High Level Summary

A solution based around a common blockchain would address the problems highlighted above. In particular, a common record of sales which is updated at the point of sale, irrefutable and immediately visible to all parties would prevent the reconciliation problems currently faced. All parties would be viewing the same data, which would be exportable into any local systems they might use.

Smart contracts would learn the claim history for each product, and adjust the price automatically in real time without the need for new ratings tables to be loaded. If challenged by regulators, the evidence justifying the increases would be readily available and transparent, clearly timestamped.

A common record of claims, and the cost of fixing problems, would also be available.

The customer would benefit by being able to find their policy easily. There should be no dispute about what was purchased as the record is tamper-proof. They also know that their personal data is protected by high-end cryptography. This protects all parties, very important in this age of the GDRP and the risk of losing up to 4% of the company’s worldwide turnover.

Retailers would benefit by prompt, automatic payment of commissions together with less time spent reconciling data. It would now be worthwhile to take the time to sell even small policies, as the downstream time is minimal.

Service agents would be able to see this and know that if they were to charge a higher price, this would be visible to everyone and need to be justified. No claims or fixes could be hidden or duplicated; payments to the service agents would be made according to the work logged in the blockchain and acceptance of that work by the customer. Data is available to everybody in real time.

Zurich benefit by:

* Reduced costs – far less time is spent on reconciliations, investigating fraud or arguing over coverage or comparative repair cost. Human touch is taken out of the day to day process, with smart contracts making the decisions around pricing and repairs.
* More business – retailers will be more willing to sell these policies if the downstream pain is removed. Better reporting will help identify customers for further marketing campaigns
* Reduced loss ratio as rating algorithms better reflect the robustness of each product
* Better reporting – reports are updated in real time, with irrefutable data that is visible to everyone
* Improved control over the repair process
* More openness and transparency for auditors and regulators, due to real time and accurate reporting, which should lead to improved brand reputation
* Peace of mind of knowing that customer data is held securely

# Our Solution

We have produced a web application which covers several scenarios.

## Sale of Extended Warranties

This happens at the Point Of Sale. The retailer searches for the customer, and if the customer isn’t found then a new customer record is created. The product itself would be filled in from the POS terminal, leaving us to add details such as the price.

## Search & Claim

This could be used by various actors. The user searches for a customer by name, and is presented with a list of matching customers from where one is selected.

The policies for that customer are then shown. One customer can have multiple policies, with there being one policy per product.

Once a policy is selected, the policy details and claims for that policy are shown.

A claim can be viewed, or a new claim added for the particular policy.

All data is written to the blockchain.

# Technology Used

Front end: **React.JS**

Service level: .**NET (C#, Web API)**

We produced our own classes based on the Etherium structure. A smart contract is a class, and a transaction chain is a list of classes embedded in the smart contract as an array.

We took the Hyperledger fabric project and based our classes on that protocol.

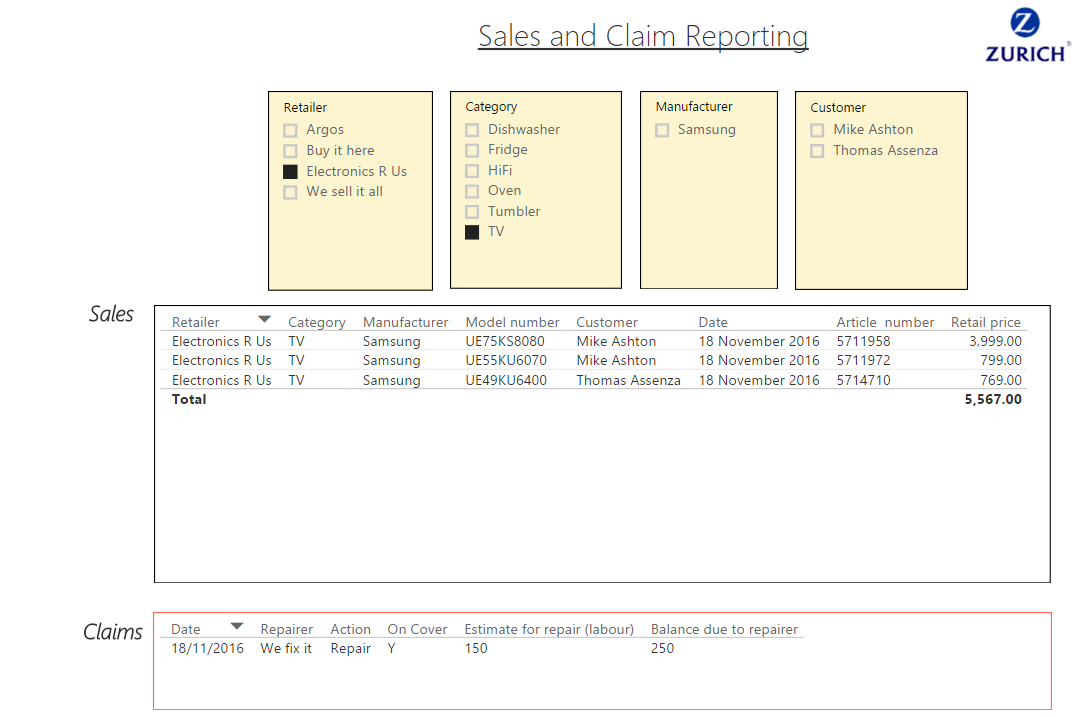
All data is stored in the service memory, there is no database.

At this point we are doing no fault tolerance, encryption or distributed ledger; this would all be needed for production level code.

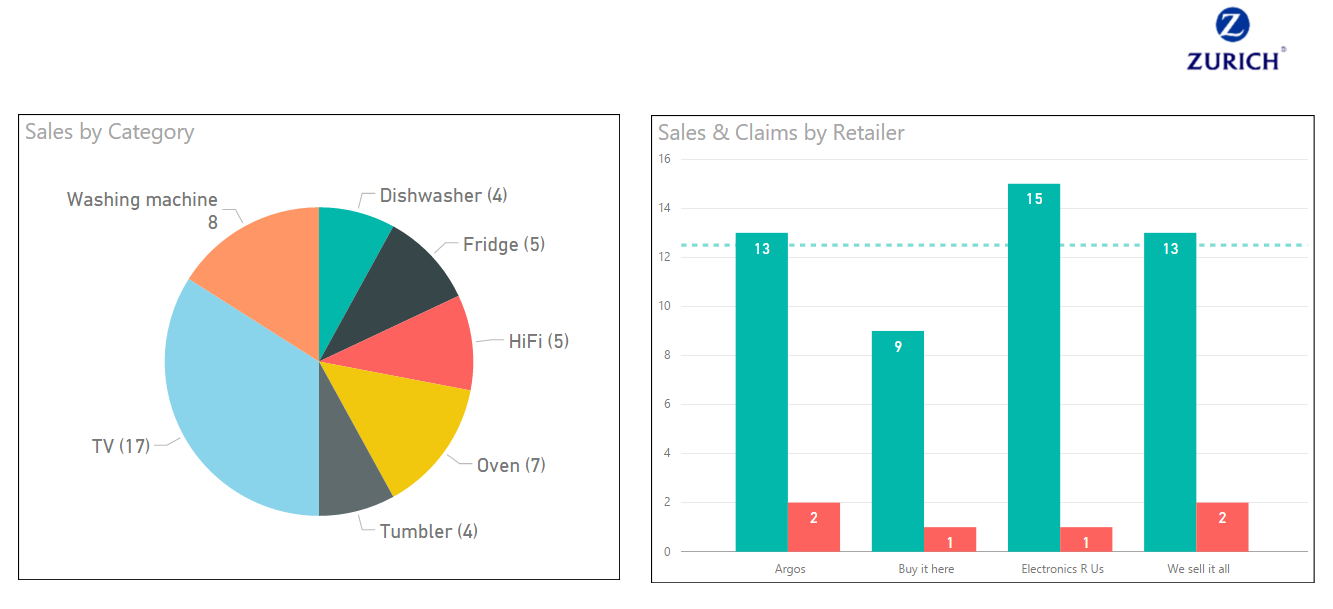
# Reporting

We produced some sample reports using PowerBI. These are currently running against test data but in the full system would be embedded into the application running against the blockchain. We probably won’t demonstrate this in the presentation as this is a Blockchain event, but the PowerBI file and the supporting spreadsheet with the test data have been loaded onto Github. You will need to download PowerBI from the Microsoft site to run the report locally.

It is easy to produce a report allowing us to filter quickly by retailer, product category, manufacturer and customer in any combination. Just click and unclick on the yellow boxes to filter dynamically.



It is also easy to show summary totals and compare sales and claims by retailer, to quickly give a visual summary of the most successful retailers.



## Team NChained

Our team comprises a mix of IT personnel from Global Aerospace, a specialise aviation insurance company based in the City of London. We have been interested in the blockchain for some time, and saw this event as an ideal opportunity to train our developers in a far more imaginative way than we would normally do.

Thank you for the opportunity to take part, and for allowing such a large contingent to descend upon you. From our perspective, we have had a very enjoyable couple of days, we’ve learned a lot, had a good team-building experience and given a clear signal to our developers that we want to invest in them and will try new ways to do this. Thank you.



Mike Ashton 19/11/2016