Blockchain Development
Week: 10
Title: Case Study

Dr Ian Mitchell

Middlesex University,
Dept. of Computer Science,
London

September 26, 2019

Lecture Aims

Aims

To gain a holistic view of a blockchain application by investigating the parts, then see how these parts combined add up to more than the whole.

Lecture Objectives

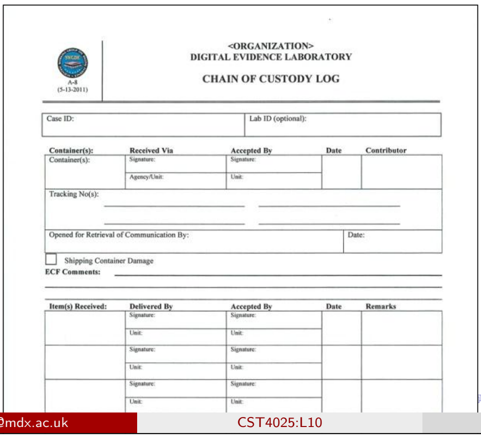
Case Study

Today's lecture will investigate the blockchain implementation of evidence management for a forensic lab, colloquially known as a chain-of-custody [1]

Chain of Custody [2]

Scientific Working Group on Digital Evidence

A-8: Chain of Custody Log



A-8
(5-13-2011)

<ORGANIZATION>
DIGITAL EVIDENCE LABORATORY

CHAIN OF CUSTODY LOG

Case ID:	Lab ID (optional):

Container(s):	Received Via	Accepted By	Date	Contributor
Container(s):	Signature:	Signature:		
	Agency/Unit:	Unit:		

Tracking No(s): ______________ ______________
______________ ______________

Opened for Retrieval of Communication By:	Date:

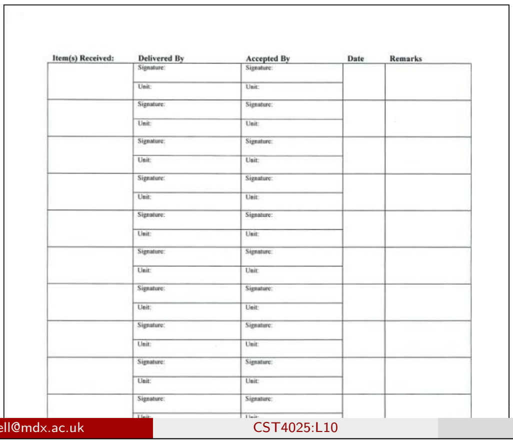
□ Shipping Container Damage

ECF Comments: ______________

Item(s) Received:	Delivered By	Accepted By	Date	Remarks
	Signature:	Signature:		
	Unit:	Unit:		
	Signature:	Signature:		
	Unit:	Unit:		
	Signature:	Signature:		
	Unit:	Unit:		

Chain of Custody [2]

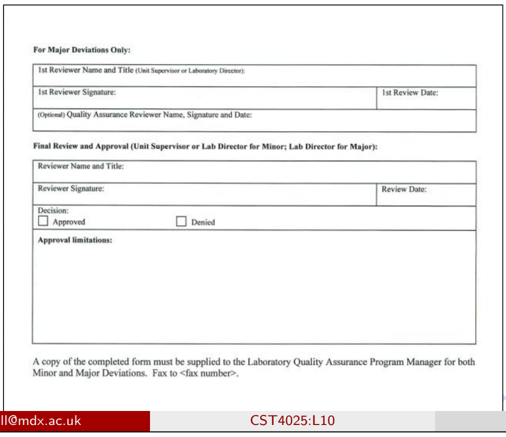
Scientific Working Group on Digital Evidence



Item(s) Received:	Delivered By	Accepted By	Date	Remarks
	Signature:	Signature:		
	Unit:	Unit:		
	Signature:	Signature:		
	Unit:	Unit:		
	Signature:	Signature:		
	Unit:	Unit:		
	Signature:	Signature:		
	Unit:	Unit:		
	Signature:	Signature:		
	Unit:	Unit:		
	Signature:	Signature:		
	Unit:	Unit:		
	Signature:	Signature:		
	Unit:	Unit:		
	Signature:	Signature:		
	Unit:	Unit:		
	Signature:	Signature:		
	Unit:	Unit:		
	Signature:	Signature:		
	Unit:	Unit:		

Chain of Custody[2]

Scientific Working Group on Digital Evidence



For Major Deviations Only:

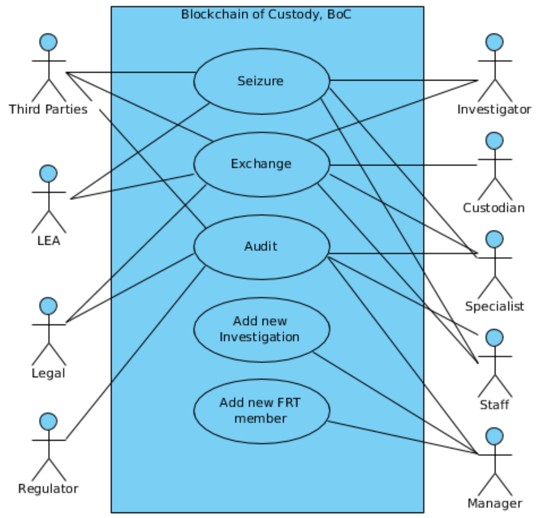
1st Reviewer Name and Title (Unit Supervisor or Laboratory Director):	
1st Reviewer Signature:	1st Review Date:
(Optional) Quality Assurance Reviewer Name, Signature and Date:	

Final Review and Approval (Unit Supervisor or Lab Director for Minor; Lab Director for Major):

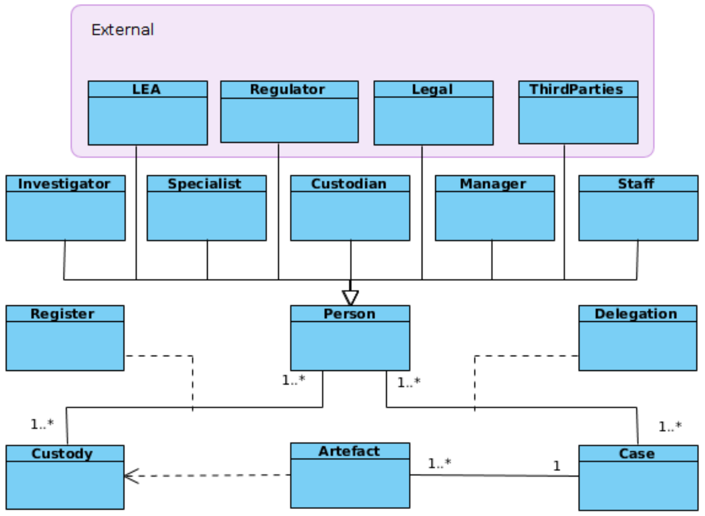
Reviewer Name and Title:	
Reviewer Signature:	Review Date:
Decision: □ Approved □ Denied	
Approval limitations:	

A copy of the completed form must be supplied to the Laboratory Quality Assurance Program Manager for both Minor and Major Deviations. Fax to <fax number>.

Use Case Diagram



Class Diagram



CTO I

Listing

```
/*
* Blockchain of Custody
* Prototype for all organisations involved in the transfer of evidence
* Single blockchain application, with the intention of becoming many inter-dependent applications
*/
namespace org.boc.net
/*
* PARTICIPANTS
*/
abstract participant Person identified by PIN{    // other personal details can be included
  o String PIN regex = /P[0-9]{1,6}/              //PIN has format P9, P99, .., P999999
  o String firstName optional                     // prototype firstName is optional
  o String lastName optional                      // prototype lastName is optional
}
participant Investigator extends Person{}         // Investigating crime, e.g., DCI
participant Specialist extends Person{}           // Forensic Specialist, e.g., Coroner
participant Staff extends Person{}                // Other internal staff
participant Custodian extends Person{}            // Custodian of evidence
participant MLA extends Person{}                  // Mutual Legal Assistance
participant Manager extends Person{}
abstract participant Organisation identified by OID{ // other data can be added
  o String OID regex = /B[0-9]{1,4}/         // B for Business, since O=0? unambiguous unique identifier
  o String OrganisationName              // name of organisation
  o String Abbr                    // abbreviate name
}
```

CTO II

Listing

```
participant Regulator extends Organisation{}    // Regulator, e.g., FSR
participant Auditor extends Organisation{}      // Audit, e.g., for ISO17025
participant Legal extends Organisation{              // Legal Professionals external to organisation
  o String email                                     // email - was candidate for unique key
}
participant TP extends Organisation{            // 3rd parties
  o String description                          // how they are related to investigation, e.g., defence
  o String email                                // communication
}
participant LEA extends Organisation{
  o String email optional                            // email
}
/*
* ASSETS
*/
asset Artefact identified by ArtefactID{    // added to investigation list
  o String ArtefactID //regex = /A[0-9]{1,5}/ // relates to unique id for evidence
  o String description                      // description of artefact, usually required
  -->Staff custodian                        // intially -->STAFF; custodian could be any participant
}


asset Investigation identified by InvestigationID{  // Case is reserved word and can be confusing
```

CTO III

Listing

```
  o String InvestigationID regex = /C[0-9]{3}/       // Case ID == InvestigationID
  --> Manager MoI                                  // Manager of Investigation, people
  o Person[] FRT                                  // list of staff eligible to seize evidence
  o Artefact[] ArtefactList                         // intially empty, populated by seizure
}
/*
* TRANSACTIONS
*/
transaction seizure{                    // initial seizure, adding an artefact
  -->Investigation currentInvestigation //investigation to update
  //-->Staff submitter                  // authorised staff
  o Artefact evidence                  // new item of evidence, concat to investigation's artefactList
}
transaction exchange{                   // exchange custody of evidence, only current owner's can swap
  -->Staff receiver                     // receiver can be custodian - needs update
  -->Artefact evidence                  // existing item of evidence
}
transaction addInvestigation{
  //o Investigation ni    // add new investigation
  o String ID   ///regex=/C[0-9]{3}/
}
transaction newMemberFRT{        // add existing member of staff to FRT
  --> Investigation currentInvestigation  // current investigation, exists
  --> Staff newMember              // member of staff to add, repeated 'n' times
```

CTO IV

Listing

```
  }

/*transaction newMemberREAD{        // add member of staff who can have read access to investigation
  --> Investigation currentInvestigation // current investigation
  --> Staff newMember              // member of staff to add to read list, can be in FRT
}
transaction newMemberALL{     //add member of staff who can have ALL access to investigation
  --> Investigation currentInvestigation // current investigation
  --> Staff newMember             // staff to add to ALL access list
}
transaction removeMemberREAD{   //remove a member of staff from the read list
  --> Investigation currentInvestigation // current investigation
  --> Staff exMember            // staff to remove from READ list
}
transaction removeMemberALL{    //remove a member of staff from ALL access list
  --> Investigation currentInvestigation // current investigation
  --> Staff exMember
}*/
```

ACL I

Listing

```
/*
* access control list for evidence network, boc
*/

rule staffSeeSelf{
    description:    "staff can only see themselves"
    participant(p): "org.boc.net.Staff"
    operation:      READ
    resource(r):    "org.boc.net.Staff"
    condition:      (p.getIdentifier()===r.getIdentifier())
    action:         ALLOW
}
rule managerSeeSelf{
    description:    "manager see self"
    participant(p): "org.boc.net.Manager"
    operation:      READ
    resource(r):    "org.boc.net.Manager"
    condition:      (p.getIdentifier()===r.getIdentifier())
    action:         ALLOW
}
rule managerSeeStaff{
    description:    "manager see staff"
    participant:    "org.boc.net.Manager"
    operation:      ALL
    resource:       "org.boc.net.Staff"
    action:         ALLOW
}
/*rule managerInvestigation{
```

ACL II

Listing

```
    description:    "manager see case"
    participant(p): "org.boc.net.Manager"
    operation:      ALL
    resource(r):    "org.boc.net.Investigation"
    condition:      (p.getIdentifier()===r.MoI.getIdentifier())
    action:         ALLOW
}*/
rule managerInvestigation{
  description: "manager sees cases"
  participant: "org.boc.net.Manager"
  operation: ALL
  resource:   "org.boc.net.Investigation"
  action: ALLOW
}
rule managerTXnewMemberFRT{
    description: "manager is allowed updates for all transactions"
    participant: "org.boc.net.Manager"
    operation:   ALL
    resource:  "org.boc.net.newMemberFRT"
    action: ALLOW
}
rule managerTXaddInvestigation{
  description: "add new investigation"
  participant: "org.boc.net.Manager"
  operation: ALL
  resource: "org.boc.net.addInvestigation"
  action: ALLOW
}
```

ACL III

Listing

```
rule managerSeeArtefacts{
   description:     "manager see case artefacts"
   participant:  "org.boc.net.Manager"
   operation:       READ
   resource:     "org.boc.net.Artefact"
   action:          ALLOW
}
rule staffSeeArtefacts{
  description:  "staff see artefacts"
  participant:  "org.boc.net.Staff"
  operation:  READ,CREATE,UPDATE
  resource:   "org.boc.net.Artefact"
  action:   ALLOW
}
rule staffUpdateArtefactList{
  //only those allowed to seize can do this
  description: "upon seizure staff need to update both artefacts and artefact list in
     investigation"
  participant: "org.boc.net.Staff"
  operation: READ, CREATE, UPDATE
  resource: "org.boc.net.Investigation"
  action: ALLOW
}

rule staffSeeHistory{
  description: "Staff can only see their own history"
  participant(t): "org.boc.net.Staff"
  operation: READ
```

ACL IV

Listing

```
  resource(v): "org.hyperledger.composer.system.HistorianRecord"
  condition: (v.participantInvoking.getIdentifier() != t.getIdentifier())
  action: DENY
}
rule staffTXseizure{
  description: "Staff can seize new artefacts"
  participant: "org.boc.net.Staff"
  operation:  ALL
  resource: "org.boc.net.seizure"
  action: ALLOW
}

rule staffTXexchange{
  description:  "staff can exchange existing artefacts"
  participant:  "org.boc.net.Staff"
  operation:  ALL
  resource:   "org.boc.net.exchange"
  action: ALLOW
}

rule SystemACL {
    description:  "System ACL to permit all access"
    participant: "org.hyperledger.composer.system.Participant"
    operation: ALL
    resource: "org.hyperledger.composer.system.**"
    action: ALLOW
}

```

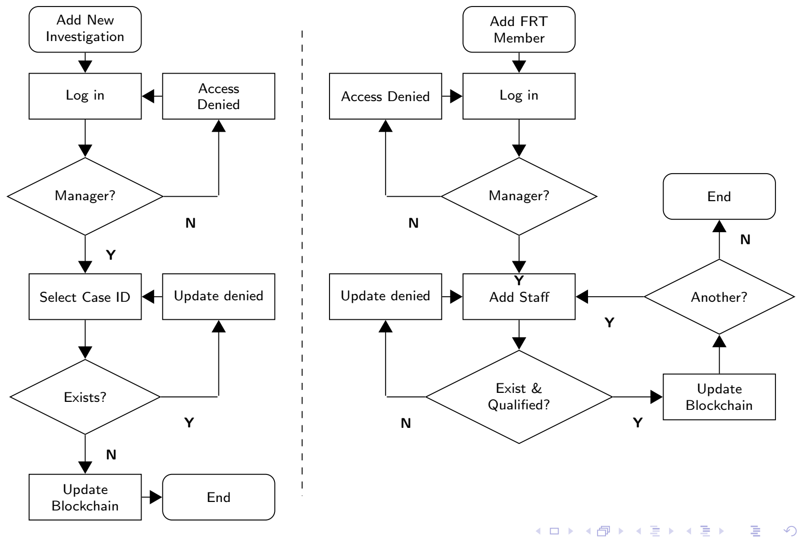
ACL V

Listing

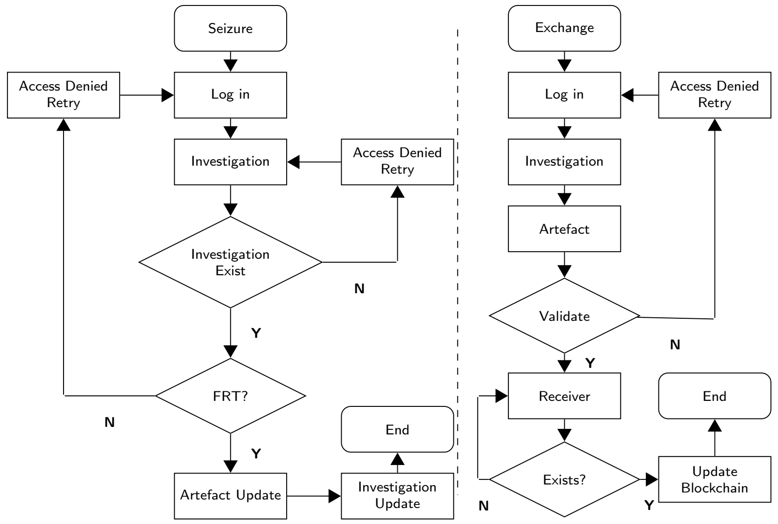
```
rule NetworkAdminUser {
    description: "Grant business network administrators full access to user resources"
    participant: "org.hyperledger.composer.system.NetworkAdmin"
    operation: ALL
    resource: "**"
    action: ALLOW
}

rule NetworkAdminSystem {
    description: "Grant business network administrators full access to system resources"
    participant: "org.hyperledger.composer.system.NetworkAdmin"
    operation: ALL
    resource: "org.hyperledger.composer.system.**"
    action: ALLOW
}
```

Adding a new investigation



Seizure and Exchange



Seizure

Access Denied Retry

Log in

Investigation

Access Denied Retry

Investigation Exist

N

Y

FRT?

N

Y

End

Artefact Update

Investigation Update

Exchange

Log in

Access Denied Retry

Investigation

Artefact

Validate

N

Y

Receiver

End

Exists?

Update Blockchain

N

Y

Logic I

Listing

```
/*
* script file for evidence network, boc
* each transactions is described below and provides the business logic


* TRANSACTIONS */

/*
* add a member of staff to FRT
* @param{org.boc.net.newMemberFRT} newMemberFRT - add member of staff to FRT list
* @transaction
*/
async function newMemberFRT(tx) {
    //check info.newMember exists as member of staff
    // shortcomming - could be a specialist or other, for now just check staff
    var ns='org.boc.net.';
    var me=getCurrentParticipant();
    return getParticipantRegistry(ns+'Staff')
        .then(function (staffReg){
            return staffReg.exists(tx.newMember.getIdentifier());
        })
        .then(function(exists){
            //if member of staff exists then update & current user is manager of
     investigation (MoI)
            // if((exists)&&(me.getIdentifier()===tx.currentInvestigation.MoI.
     getIdentifier()))
            // current manager is check by ACL, no need to repeat
            // if member of staff exists
```

Logic II

Listing

```
            if (exists)
            {
                tx.currentInvestigation.FRT = tx.currentInvestigation.FRT.concat(tx.
     newMember);
                return getAssetRegistry(ns+'Investigation')
                    .then(function(investigationReg){
                        return investigationReg.update(tx.currentInvestigation)
                    })
            } else {
                throw new Error('Staff member'+tx.newMember.getIdentifier()+' does not
     exist! ')
            }
        })
    }

   /*
   * add a new investigation
   * @param {org.boc.net.addInvestigation} addInvestigation - add a new investigation,
     with MoI
   * @transaction
   */
  async function addInvestigation(tx){
      //acl means only managers can view their own cases
      // does not check if case number already exists
      var ns='org.boc.net.';
      var me=getCurrentParticipant();
      return getAssetRegistry(ns+'Investigation')
        .then(function (investigationReg){
```

Logic III

Listing

```
                var factory = getFactory();
                var ni = factory.newResource('org.boc.net','Investigation',tx.ID);
                ni.MoI=me;
            //ni.Authority='';
            ni.ArtefactList=[];
            ni.FRT=[];
                return investigationReg.add(ni);
        })
  }
/*
* add a new artefact
* @param {org.boc.net.seizure} seizure - add a new artefact
* @transaction
*/
async function seizure(tx){
    var ns='org.boc.net.';
    var me=getCurrentParticipant();
   return getAssetRegistry(ns+'Investigation')
        .then(function(iReg){
           //check investigation exists
            return iReg.get(tx.currentInvestigation.getIdentifier())
                .then(function(singleInvestigation){
                //check current user is member of FRT exists
                    var needle=me.getIdentifier();
                    var haystack=singleInvestigation.FRT;
                    var filteredHaystack = haystack.filter((item)=>item.PIN===needle);
                    if (filteredHaystack.length>0){
```

Logic IV

Listing

```
                        //cannot submit on behalf of others, default option of ownership is me
                        tx.evidence.custodian = me;
                        singleInvestigation.ArtefactList = singleInvestigation.ArtefactList.concat(tx.evidence);
                        iReg.update(singleInvestigation);
                        //update artefact
                        return getAssetRegistry(ns+'Artefact')
                          .then(function(aReg){
                            return aReg.add(tx.evidence);
                            });
                    } else {
                        throw new Error('User is not a member of FRT for this investigation')
                    }
                })

        })
}

/*
* exchange an existing artefact
* @param {org.boc.net.exchange} exchange - exchanging the custodian of an existing artefact
* @transaction
*/
async function exchange(tx){
    const me=getCurrentParticipant();
```

Logic V

Listing

```
    //staff cannot update on behalf of others
    if(me.getIdentifier()!==tx.evidence.custodian.getIdentifier()){
        throw new Error('Staff cannot update on behalf of other staff');
    } else {
            // artefact is in artefactList, since it has been submitted
            tx.evidence.custodian = tx.receiver
            //update
            let assetRegistry = await getAssetRegistry('org.boc.net.Artefact');
            await assetRegistry.update(tx.evidence);
    }

}
```

Output

```
{
 "$class": "org.boc.net.addInvestigation",
 "ID": "C111",
 "transactionId": "ae86b543-54a4-444c-8aeb-28268cf02ff5",
 "timestamp": "2018-08-28T15:57:21.212Z"
}
{
 "$class": "org.boc.net.addInvestigation",
 "ID": "C222",
 "transactionId": "c8b05472-9299-4579-9c34-2788b4a731d7",
 "timestamp": "2018-09-04T09:53:00.558Z"
}
```

Summary

Conclusions

Chain of Custody

References I

[1] I. Mitchell et al. “Blockchain of Custody, BoC”. In: *Cyber Security Practitioner’s Guide* (2019). Ed. by Hamid Jahankhani.

[2] Scientific Working Group on Digital Evidence (SWDGE). *Model Standard Operation Procedures for Computer Forensics (ver. 3).* https://www.swgde.org/. Version 3. [Accessed June 2016].

Web Resources

- `http://hyperledger.org`
- `https://nodejs.org`