Poseidon House
Castle Park
Cambridge CB3 0RD
United Kingdom

TELEPHONE: **Cambridge (0223) 323010**
INTERNATIONAL: **+44 223 323010**
FAX: **+44 223 359779**
E-MAIL: **apm@ansa.co.uk**

ANSA Phase III

A dependability scenario — IPS (Information Publishing System)

Nigel Edwards, Ed Oskiewicz, Owen Rees

Abstract

This document describes the requirements and design of a distributed information publishing system. The intention is to design a system which raises a number of dependability issues. Accordingly, several aspects of the system which do not raise dependability issues are ignored.

APM.1096.00.02 **Draft** 23 December 1993

Request for Comments (confidential to ANSA consortium for 2 years)

Distribution:
Supersedes:
Superseded by:

Contents

1 Problem statement and initial analysis

This chapter gives a problem statement, data dictionary (responsibility model) and object model for a scenario for the ANSA dependability group. The application is a distributed information publishing system. The intention is to focus on the dependability requirements, other issues such as federating different information models are not addressed in detail. Some of the excluded issues are summarised in §1.5.

1.1 Structure of document

This chapter contains the initial problem statement, the object model and responsibility model using the methodologies described in [RUMBAUGH 91] and [SHARBLE 92]. Chapter 2 presents the dynamic model and chapter 3 presents the functional model, both according to the methodology presented in [RUMBAUGH 91]. From the problem statement it will be apparent that IPS is a user interface application, driven by the activities of customers and authors. Hence the dynamic model and object model dominate the analysis, the functional model presented in chapter 3 is less important [RUMBAUGH 91].

1.2 IPS an Information Publishing System

The objective is to develop a distributed information system in which different **organisations** can **publish** information electronically using hypermedia.

Scoop Inc., a market research company, is a typical organisation which wants to publish **information**. **Customers** should be able to **browse** through what is available. The **charges** for browsing the information will be related to its value. So a small charge (nothing) is made for browsing the top level description of what is available. Viewing more detailed (and valuable) information is more expensive. Each unit of information has a cost associated with it.

The information is the company's primary source of revenue, so it is important that it is highly available (for example, 1 minute per year or less of unplanned unavailability). In a global information market it is not possible to take down information servers to reconfigure them: somebody somewhere will want to use it (and pay for it). This means they need to be available 7 days a week 24 hours a day, so there is no planned unavailability.

Integrity of the information is also vital it must not be corrupted under any circumstances. The billing information must not be lost and must relate accurately the information which the customer has accessed. Some of this information may be maintained or available from the systems environment (e.g. account databases), other information must be maintained explicitly by the system itself.

Customers have their own contexts called **sessions**. This means that customers do not have to start from the same place each time they use the system — the session remembers where they were. Customers can suspend and resume sessions and may or may not be at the same site when a session is resumed. Customers may or may not be charged for each access of a particular piece of information within a session. This depends on the charging policy associated with that information.

Customers can also make printed copies of textual information on their own site. They will be charged for each copy.

Customers can also make electronic **copies** of the information. Rather than charging for copying, the (much simpler) alternative used here accepts that you cannot really control copying, but charges on the basis of use. The local copy does not allow itself to be read unless it can log charging information. This is the approach advocated in [MORI 90]. One security issue which needs solving is how to prevent customers copying information, editing it and then publishing it as their own.

Note: Is it possible to to prevent anybody accept the owner editing information?

Some customers will have **accounts** with Scoop, so charges will be incurred to these accounts. Other customers will pay by providing access to a source of revenue (e.g. charge or credit card number + PIN). Authorisation and billing to the relevant **card company** is on-line.

Other **organisations** also publish information. Some of Scoop's information may have references to information provided by others. If a customer moves into another organisation's system, Scoop is not entitled to charge them for this information, but the information owner may wish to receive some form of payment.

Note that although Scoop is not entitled to charge for information owned by another organisation, it has charged for the information it owns which points at other information in the system. So one possible service Scoop could offer is a directory service which points at information owned by other organisations. Access to the directory would be chargeable.

Charging needs to be made on the basis of the identity of the person accessing the information. It would be unusual for the owner of information to be charged for accessing it.

Facilities need to be available for **publishing** information and updating or withdrawing existing information. Note that when information is updated, it may be referenced and will have references to other information. These references need to be kept up to date.

The following already exists:

- The card company databases
- Some of the information, but new information will be continually added and existing information will be continually updated.

What constitutes information could vary from a being the size of a book down to a report of the latest share price. Information may be in the form of various kinds of media: text, video or audio. Large units of information will be updated infrequently, small ones may be updated very frequently (order of minutes). Very new, information may be charged at a premium rate: the fresher it is the more it is worth.

Some customers need to be **notified** when information is updated (e.g. share price). This taken together with the notion of customers having sessions, allows customised information packages to be built. For example, a business analyst specializing in the pharmaceutical industry would have the latest press reports and share prices of the industry available. Certain events (e.g. rapid share price change or year-end figure publication) might trigger alarms to bring it to the analyst's attention. Use of the notifier is chargeable.

1.2.1 Discussion

The above problem statement contains several issues which are generic service management problems:

- Publishing information
- Controlling its availability and consistency
- Charging for accessing it and maintaining the integrity and availability and consistency of the charging information
- Controlling copies

The information units can be regarded as network elements.

The World Wide Web (WWW) already has some of this functionality: it is a distributed hypermedia system. In the WWW it is common for information services to go off the air and bounce requests. There is no attempt to automatically maintain consistency: it is possible to find that a hyperlink is pointing at outdated or unavailable information.

This application turns information into commodity which is bought and sold electronically. This means that dependability becomes very important especially in addressing issues of vendors charging for information, updating information (possibly frequently) and maintaining the consistency of hyperlinks and a rapidly changing dynamic environment. These are issues which are not solved in the worldwide web (but then that is free).

By including audio and video and allowing customised information packages to be built, this scenario subsumes the video on demand and "ME-TV" concepts which are being discussed by the industry. "ME-TV" is analogous to putting together your own information package using (video) information which is published by other suppliers.

1.2.2 Scope of the problem

The reason of designing this system is to understand the dependability issues. Hence there are a number of issues which will not be discussed:

- Federating different information models — different organisations may have different formats for their information
- Performance is assumed to be adequate
- Actual payment — customers may want the facility to receive one bill for all the information accessed and have their payment disbursed to the information owners
- The initial analysis (the object model) will show a system containing audio and video information as well as text, for the more detailed analysis (the functional and dynamic models) we will only consider text

1.2.3 Services available now

A brief description of some information systems for business news which are available now appear in [SHILLINGFORD 93]:

- Reuter Company Newsyear — a newswire providing real-time news structured according to business categories e.g. telecoms, petro-chemical, pharmaceutical, banking, insurance, automobile, energy, environment, etc.
- London Stock Exchange's Topic database — latest share movements
- Reuter Textline — a database of multi-national corporations
- Datastream — current share prices
- FT-Profile — a database of business information

The charging models for these information systems is very simple: you rent a lease line and pay a subscription. There is no notion of hyperlinks, notification or customising "information packages". Hence the service management problems are much simpler.

Note: The statement on charging model is based on very limited information.We would appreciate more information.

Other information services are also available on a commercial basis. For example, press agencies such as UPI sell news, and this can be automatically fed into a searchable information service with restricted access.

1.3 Classes: responsibility and data dictionary

This section identifies some classes and lists their responsibilities using the methodology suggested in [SHARBLE 92]. A number of attributes and other associations (not covered by responsibility) are also identified. Taken together this section constitutes the data dictionary described in [RUMBAUGH 91]. The words highlighted in §1.2 indicate several potential classes. These classes are listed in figure 1.1. Initially we had 23 classes, but iterations of the analysis reduced the complexity to the twelve classes listed.

Figure 1.1: Classes

informationUnit	**charger**	**accounts**	**organisation**
customer	**printedUnit**	**session**	**publisher**
browser	**copiedUnit**	**cardCompany**	**notifier**

1.3.1 Information unit

An informationUnit is the basic unit of information in the system. It may contain text, graphics, video or audio. An informationUnit may contain references to other informationUnits.

Note: It is not clear whether an informationUnit may contain a mixture of types, or whether these should be considered separate informationUnits. The questions of charging and the possibility of partial failures of mixed information need to be studied. More notes on this in §1.3.3.

Note: Where do active resources like fill-in forms and searchable indices fit?

An informationUnit has a value which is assigned by its owner, this is captured in a charging policy. Some informationUnits will contain an index of what is available. There is therefore no need for any special gateway into the system. InformationUnits are owned by an Organisation.

Table 1.1: Responsibility of InformationUnit

Responsibility	Collaborator
Integrity & availability of information	
access	browser, charger, session
making copies	copiedUnit, browser
printing	printedUnit, charger, browser

1.3.2 Customer

Customers have accounts or some means of payment. They also have sessions: current sessions and past sessions which they may wish to resume[1].They are not directly responsible for anything. It might be argued that the customer is responsible for such things as payment. This is delegated to the browser. The purpose of the customer class is to show associations with other objects, such as ownership of sessions

Table 1.2: Responsibility of customer

Responsibility	Collaborator

1.3.3 Browser

The browser is the customer's interface to the system.

It can access information units specified by links in information units that have already been accessed.

It can make copies of informationUnits. It may or may not follow links to nested informationUnits during copying, depending on the customer's preferences.

Table 1.3: Responsibility of browser

Responsibility	Collaborator
Providing payment details	charger
Accessing informationUnits	InformationUnit, charger, session
Accessing CopiedUnits	CopiedUnit, charger, session
suspending session	informationUnit, session
resuming session	informationUnit, session, charger
copying	InformationUnit, copiedUnit
printing	informationUnit, printedUnit, charger

Note: Place holder for dependability issues: The browser may need to suspend or abort the accessing of an information unit. It may need to prove the access did or did not take

1. If customers are allowed to have multiple sessions then there may be a garbage collection problem, depending on where the sessions reside. How does one decide a session can be discarded?

place in the event of a dispute. The general principle is that charging needs to accurately relate what actually takes place and needs to be auditable.

Note: For nested information units, we need to consider 'contained' and 'referenced' parts, and the extent to which author and viewer control the distinction. For example, inline images in HTML (the page description language for WWW) are more 'contained' than 'referenced' but the browser can control whether or not they are retrieved together with the page. If informationUnits can be structured and retrieved in parts, this increases the scope for partial failures.

Note: Are the charging granularity and access granularity necessarily the same? Is this the real issue behind the ‘contained’ vs ‘referenced’ question?

1.3.4 Charger

The charger registers charges as information is used by customers. Charges are paid to the owner of the information. A charger consults the current session and charge policy to determine whether each access should be charged.

Table 1.4: Responsibility of charger

Responsibility	Collaborator
availability & integrity of charges	
registering charges	accounts, informationUnit, cardCompany, session, browser, notifier, copiedUnit

1.3.5 Copied unit

A copiedUnit is personal version of a informationUnit for a customer. The reasons for copying are caching to improve performance or to avoid paying network charges each time the information is accessed. A customer may choose to prepay for a given number of accesses to a copiedUnit to avoid having each access verified. Alternatively a customer may have a credit limit which allows a limited number of accesses before charges are registered.

Table 1.5: Responsibility of copiedUnit

Responsibility	Collaborator
security of copy	
Integrity & availability of InformationUnits	
Access to CopiedUnits	browser, charger, session

Note: There are ownership issues associated with copiedUnit. What happens if the information provider goes bust just after I have paid for my copy. If I own my copy I ought to have the right to see it, but if it is only 'rented' there are interesting questions to answer.

1.3.6 Account

The customer account information is maintained by accounts. Accounts are owned by organisations which publish information.

Table 1.6: Responsibility of accounts

Responsibility	Collaborator
Maintaining availability and integrity of all customer bills.	

Table 1.6: Responsibility of accounts

Responsibility	Collaborator
Updating bills	charger

1.3.7 Session

Sessions are owned by customers. A customer can resume or suspend a session. Sessions need to remember which informationUnits have been accessed, so that the charging policy can be adhered to. This information needs to be highly available: customers may have been charged a lot of money for information which they are entitled to access without further charges.

Table 1.7: Responsibility of session

Responsibility	Collaborator
integrity and availability of list of informationUnits accessed	
logging informationUnits accessed	informationUnit
resuming session	browser
suspending session	browser
logging fees paid	charger
logging copiedUnits accessed	copiedUnits

1.3.8 Card company

A cardCompany is expected to pay the information owner according the charges the customer has accumulated. This payment is outside of the system.

Table 1.8: Responsibility of cardCompany

Responsibility	Collaborator
Accepting customer billing information	charger

1.3.9 Organisation

The information provided may well have references to sources of information provided by other organisations. Organisations are ultimately responsible for the integrity and availability of the information which they publish. In addition they are responsible for pricing and collecting charges for accessing that information.

Table 1.9: Responsibility of organisation

Responsibility	Collaborator
integrity & availability of informationUnit	informationUnit
pricing information	informationUnit, publisher
publishing information	publisher
withdrawing information	publisher
updating information	publisher
collecting charges	charger, accounts, cardCompany

1.3.10 Publisher

The publisher is responsible for introducing information into the system. This means updating existing information (at least the indexes) so that they have links to the new informationUnit. A publisher must also assign a cost and owner to the information. Publishers can also update (replace) existing information with new information this may mean changing the owner or cost as well as the content. Finally publishers can withdraw information. Publishers need to preserve the integrity of cross-references between information: changes need to be atomic.

Table 1.10: Responsibility of publisher

Responsibility	Collaborator
assigning ownership	informationUnit, organisation
pricing information	informationUnit, organisation
publishing information	organisation, informationUnit(s)
withdrawing information	organisation, informationUnit(s)
updating information	organisation, informationUnit(s)

1.3.11 Notifier

The notifier is responsible for notifying sessions who have registered an interest in an informationUnit of significant events in the lifecycle of that information unit. Charges are registered for its service. For example, the informationUnit has been updated or withdrawn. The notifier is responsible for maintaining the consistency and integrity of the database of registered interests, and ensuring that the notifications are reliably delivered.

Table 1.11: Responsibility of notifier

Responsibility	Collaborator
consistency & integrity of registration data	
registering interests	session
notifying significant lifecycle events	informationUnit, publisher, session, charger

1.4 The Object Model

This section presents the object model of the information system using the notation of [RUMBAUGH 91]

Figure 1.2: Object model: Browser/InformationUnit

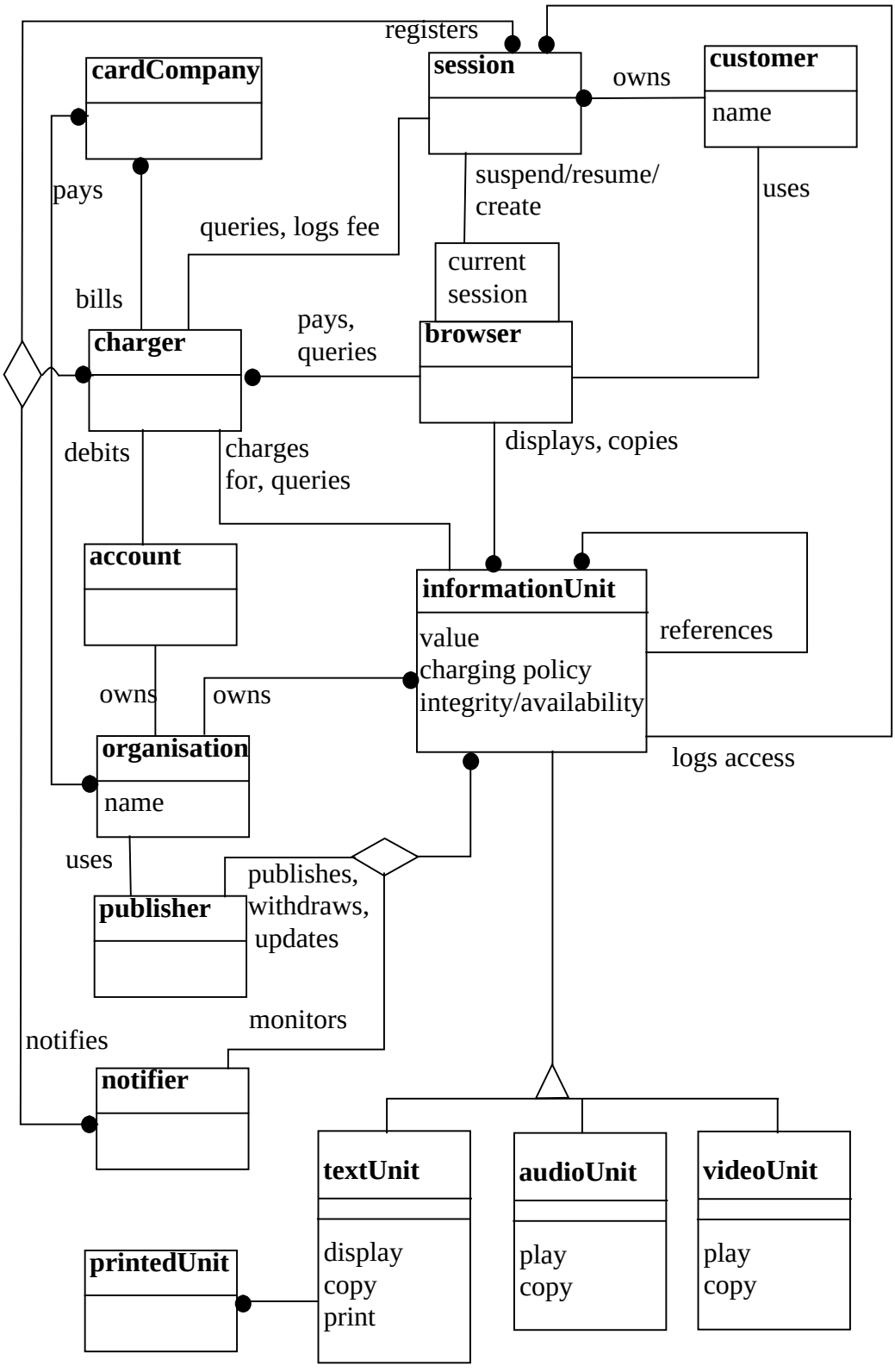
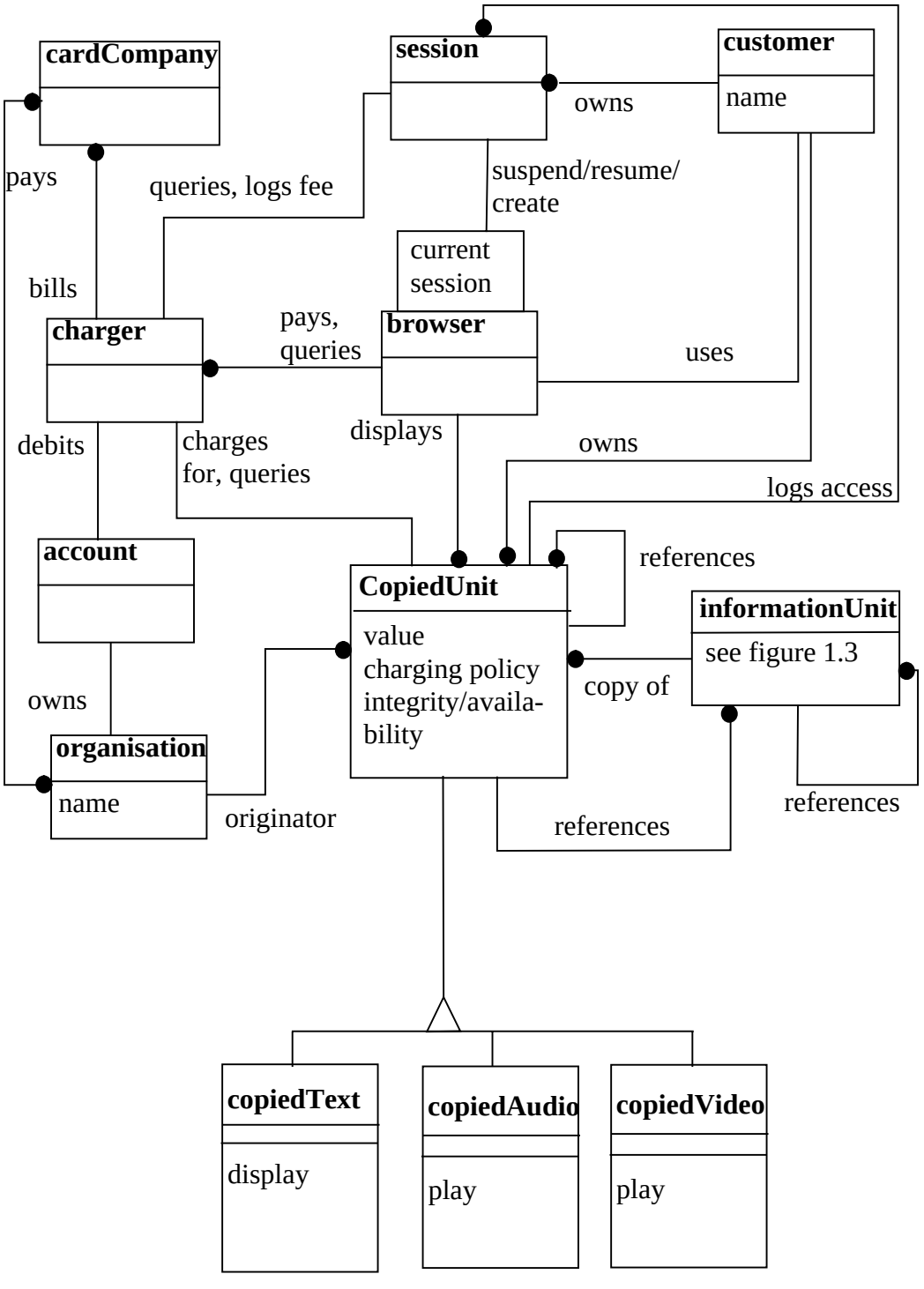


Figure 1.3: Object model: Browser/CopiedUnit



1.5 Extending the design

Note: If the objective was to design and build and information service, we would pursue the following issues. There are no obvious extra dependability issues.

The provision of an information service can be broken down into a number of pieces which could be provided by separate organisations.

- Basic data provision; gathering and making available the raw data.
- Navigation (discovery); linking related pieces of data.
- Presentation (visualisation); formatting (textual or graphical)
- Summarising; producing abstracts or digests
- Billing; keeping the accounts, collecting and distributing the money.

The scenario is based on one organisation doing all of these things. There is a passing mention of other organisations, but the possibility of Scoop Inc. getting commission for pointing a customer at another provider is not mentioned.

2 Dynamic model

This chapter presents a dynamic model of IPS using the methodology described in [RUMBAUGH 91]. First a number of system scenarios are needed which describe a typical interaction within IPS. After the first iteration of the analysis we discovered that some of the scenarios added only limited additional information. These are included in appendix A, for completeness.

In chapter 1 there was some discussion about having both internal accounts and on-line charging to credit card companies. The first attempt to build the dynamic model showed that including credit card companies added no more dependability issues which are not already included by the accounts database. The addition of the credit card facility described in chapter 1 would add many security and federation issues which we are not concerned with.

Note: This scenario shows that it is difficult to isolate security from dependability. For example, protecting the integrity of the billing data has aspects of security as well as reliability and availability, and it is difficult to make a distinction between them.

2.1 System Scenarios

2.1.1 Starting a new session and accessing an informationUnit

The event trace for this scenario is shown in figure 2.1.

1. A customer clicks on the browser icon the display gives him the option of starting a new session or resuming an old session.
2. He elects to start a new session.
3. The browser starts a creates a new session and presents the customer with a “home” informationUnit. This is essentially a directory with hyperlinks to preferred information providers (if the number of providers was large quite how this was managed would be an issue — we ignore this).
4. The customer clicks on the hyperlink to Scoop.
5. The browser consults the charger for Scoop’s top level informationUnit.
6. The charger asks the informationUnit for its cost and charging policy, it finds there is no charge for this informationUnit.
7. Because there is no cost, the charger immediate logs the null fee with the customer’s session and tells the informationUnit that the browser’s access is authorised.
8. The browser indicates that the informationUnit is free and asks the customer for permission to proceed.
9. The customer says ok, so the browser asks the informationUnit to display itself.

10. The customer notices that scoop has just published a new report on “Probable and Possible Directions in Home Entertainment” and clicks on the hyperlink.
11. The browser obtains the reference to the report's interface and its charger interface.
12. The browser accesses the charger and finds that it will cost $10.00 to read the report.

Note: The issue of how much the network vendor would charge is ignored.

13. The charger consults the customer's session and finds that the informationUnit has not been accessed before.
14. The charger tells the browser it will cost $10.00 to read the report.
15. The customer decides that the charge is ok and tells the browser to display the report.
16. The browser asks the customer for the payment method and is told to charge it to an account.
17. The browser asks the customer for a pin number and account name.
18. The browser asks the customer if the pin and account name should be cached, so that it will not be requested again in this session. The customer says yes.
19. The browser passes this information to the charger along with a reference to the new session.
20. The charger verifies the details and debits $10.00 from the customer account.
21. The charger then tells the informationUnit that the appropriate fee has been paid and logs this information into the customer's session.
22. The browser asks the informationUnit to display itself and receives a stream of text.

Note: what about the notification service? Having accessed this information the customer might want to be notified if it is updated.

2.1.2 Returning to an informationUnit within an existing session

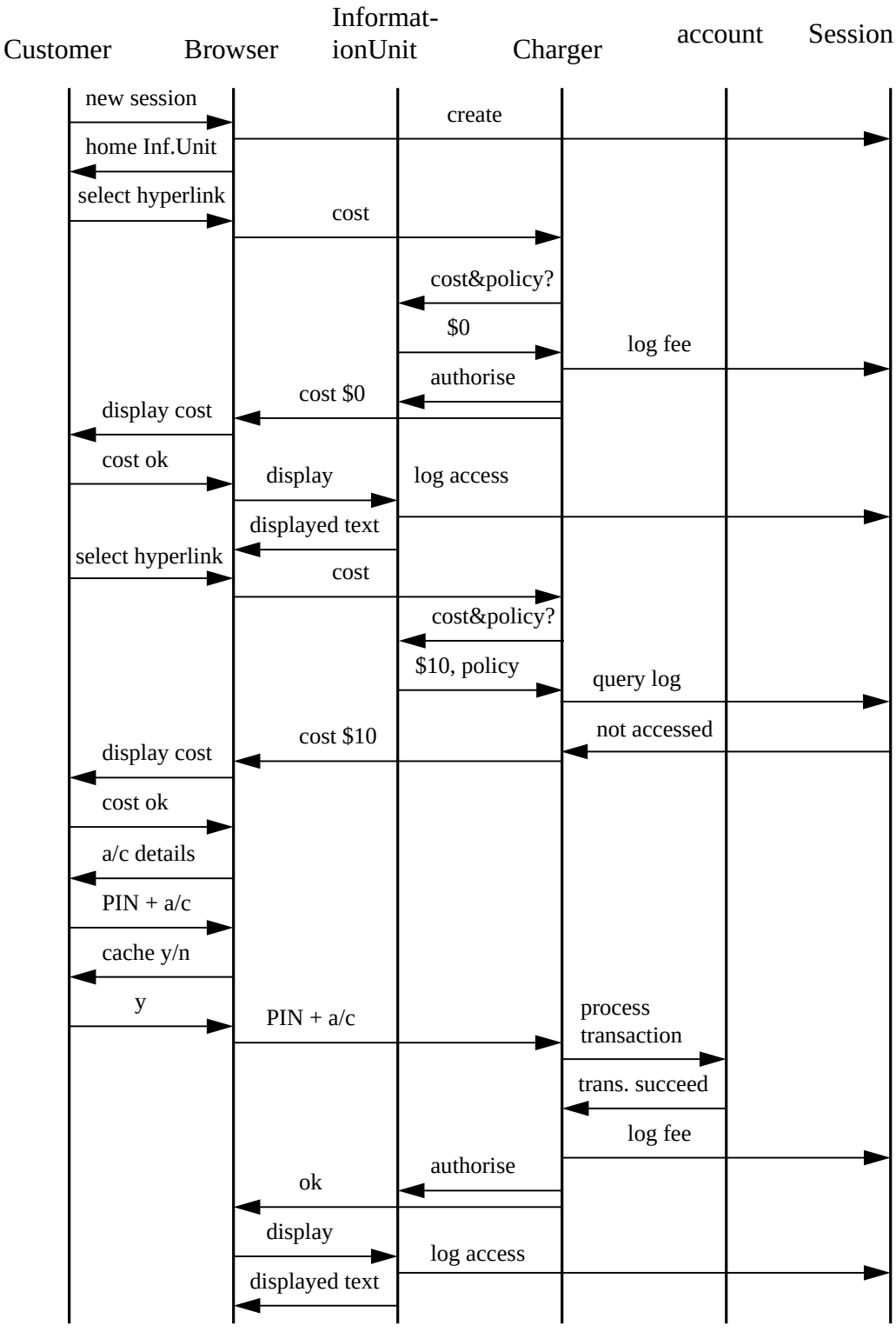
Figure 2.2 shows the event trace for this scenario.

The customer notices that Scoop's report mentions that Bogus Enterprises Inc. have announced the development of 5D television and points to an item published by NoNews describing the company's claims for the product. The customer clicks on the hyperlink to read this article (see scenario described in §A.1.1).

1. The customer finishes reading the article about 5D television and clicks on the browser to return to the original article by Scoop.

Note: In WWW this is sometimes implemented as a “back” button, alternatively the whole session could be displayed as a web. Each node would represent and informationUnit which had been visited since the session was created. The node itself would be a hyperlinks; by clicking on a node the hyperlink would be accessed. Hence the session could be represented as an informationUnit. This is explored a little further in the functional model presented in chapter 3.

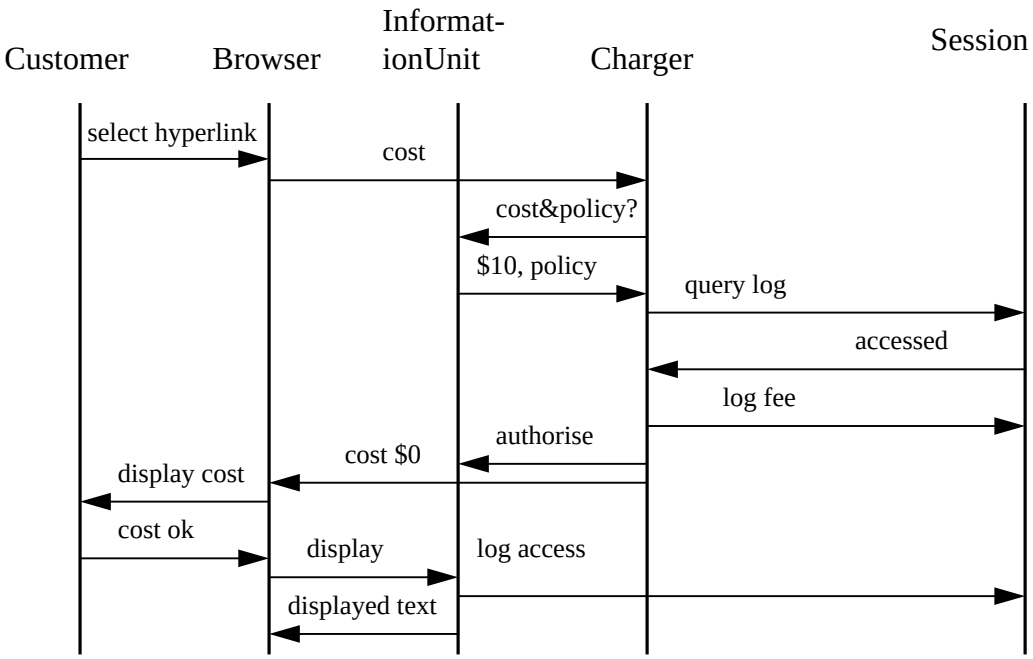
Figure 2.1: Event trace for §2.1.1



2. The browser obtains the reference to the report's interface and its charger interface from the customer's session.The browser accesses the charger to find out how much it will cost to read the report again.
3. The charger queries the informationUnit to find that the charging policy allows multiple accesses within the same session for a single charge.

4. The charger queries the customer's session to find it has already accessed this informationUnit.
5. The charger tells the browser that there will be no charge for the information; it tells the informationUnit that the appropriate fee has been paid and logs the null fee to the customer's session.
6. The browser tells the customer that there will be no further charge for the informationUnit.
7. The customer decides that this is ok and tells the browser to display the report.
8. The browser asks the informationUnit to display itself and receives a stream of text.

Figure 2.2: Event trace for §2.1.2

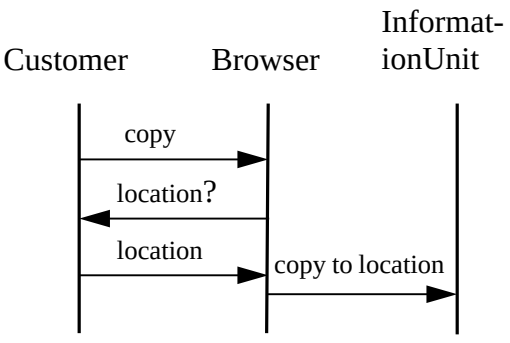


2.1.3 Copying an informationUnit

What about the notification service — the customer might like to be notified if the original is updated.

Figure 2.3 shows the event trace for this scenario.

Figure 2.3: Event trace for §2.1.3



The customer decides to make a copy of the report by Scoop on “Probable and Possible Directions in Home Entertainment”.

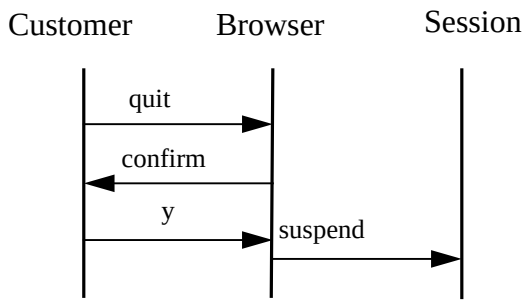
1. The customer clicks on the browser’s copy button
2. The browser asks the customer where the copy should be placed (machine, directory?)
3. The browser tells the informationUnit to copy itself to the location specified by the customer.

There is no charge for the copying of an informationUnit: only access to information is charged. To make the copiedUnit available to the outside world the customer may update and exiting informationUnit (perhaps internal to her organisation) which advertises what it available (see §2.1.5).

2.1.4 Ending a session

Figure 2.4 shows the event trace for this scenario.

Figure 2.4: Event trace for §2.1.4



1. The customer clicks on the browser button, labelled “Quit”.
2. The browser asks for confirmation.
3. The browser closes down and tidies up any information being displayed.
4. The browser shutdowns finishes off any interaction in which it is engaged (e.g. chargers)
5. The browser makes sure that the session is in stable storage.
6. The browser terminates itself.

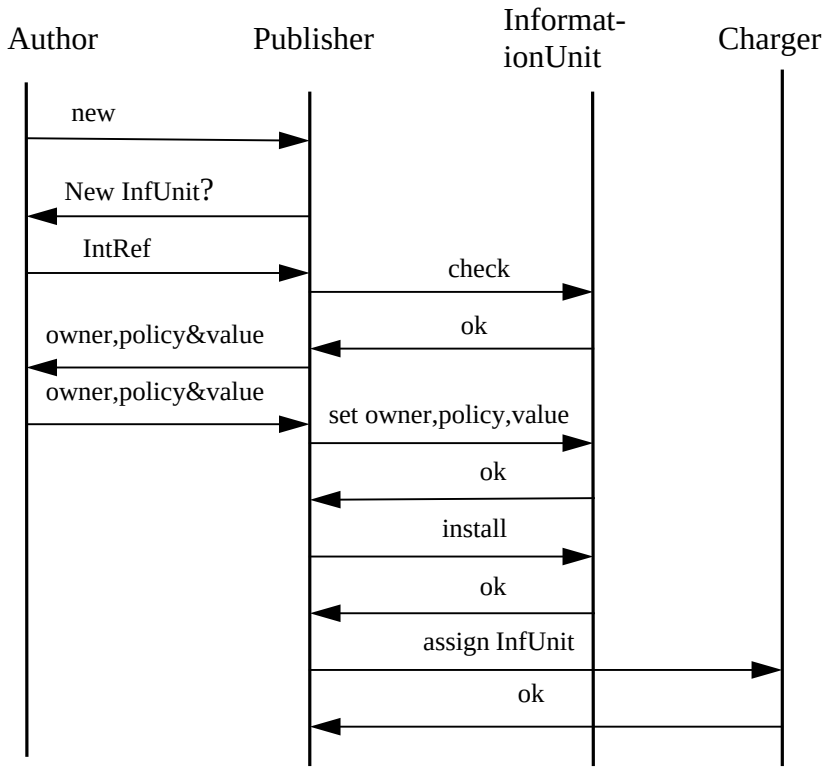
2.1.5 Publishing a new informationUnit

For the sake of simplicity, at present, this scenario considers the publication of text only. This scenario also ignores the issue of tools to assist in the preparation of a report.

The event trace for this scenario is shown in figure 2.5.

1. Jess has written a new report which she has clearance from her boss to publish on behalf of HotBox — her company.
2. She starts up the publishing tool.
3. The publisher asks her whether she wants to update an existing informationUnit or publish a new one.
4. She clicks on the button marked “new”

Figure 2.5: Event trace for §2.1.5



5. The publisher asks her for the new informationUnit (interface reference)
6. She gives the publisher the interface reference.
7. The publisher then checks the informationUnit. This includes: a format check (e.g. so it will display properly); a check that the informationUnits pointed at by its hyperlinks exist.
8. The publisher then engages in a dialogue with Jess to fix the owner, charging policy and value of the informationUnit.
9. The publisher installs the informationUnit on suitably dependable site. This includes assigning it a charger.
10. Having successfully installed the new informationUnit it is not yet referenced, so nobody can find out about it. Jess' next task is to update HotBoxes index page(s) (see §2.1.6).

Note: I don't know what "suitably dependable means" this needs sorting out — volunteers please!
The publisher would doubtless run various checks on HotBox, such as whether or not it was a bona fide organisation and had suitable means for receiving payment. This is not addressed. Would it add any new dependability issues? There are some thorny security issues which are ignored.

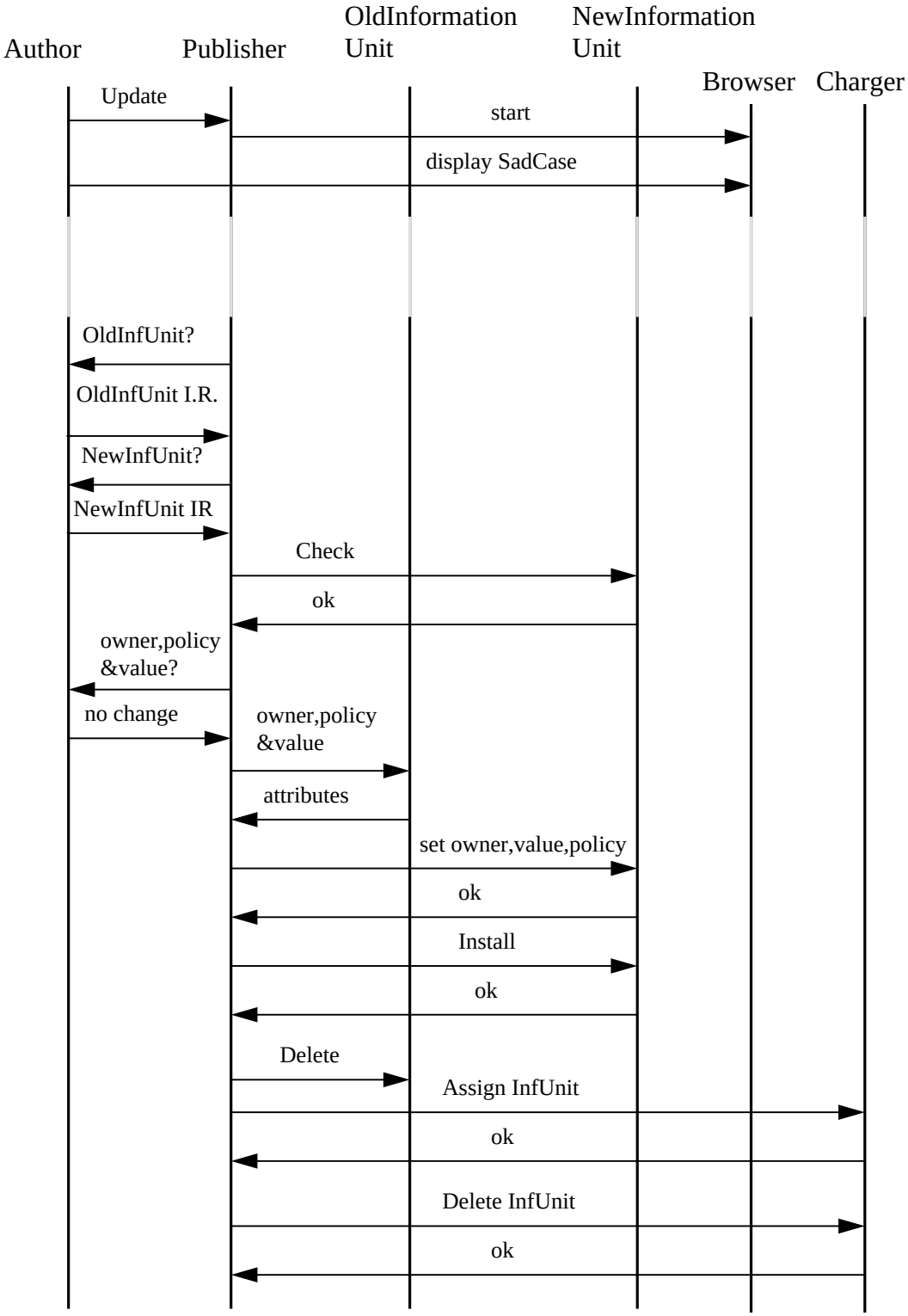
2.1.6 Updating an existing informationUnit

Updating an informationUnit involves replacing an InformationUnit with another. We ignore the provision of tools which allow somebody in the owning

organisation to copy and make changes to the copied version of the informationUnit.

The event trace for this scenario is shown in figure 2.6.

Figure 2.6: Event trace for §2.1.6



1. Jim has created an updated version of an informationUnit to replace and existing informationUnit owned by SadCase — his company.
2. He starts up the publishing tool.
3. The publisher asks him whether he wants to update an existing informationUnit or publish a new one.
4. He clicks on the button marked “update”
5. The publisher starts a browser which enables Jim to navigate through SadCase’s informationUnit’s until he locates the one he wants to change.
6. The publisher asks him for the new informationUnit (interface reference)
7. He gives the publisher the interface reference.
8. The publisher then checks the informationUnit. This includes: a format check (e.g. so it will display properly), that the informationUnits pointed at by its hyperlinks exist.
9. The publisher asks him if he wants to change the value or charging policy.
10. He says no.
11. The publisher installs the new informationUnit on suitably dependable site, overwriting the existing one.
12. This does not effect anybody who is currently accessing the existing version.

Note: Issues here: how to prevent indefinite delay because a punter is locking up the old informationUnit (and paying for it). Security issues have been ignored.

13. As soon as the update takes place the notifier updates all sessions which have registered an interest in the informationUnit.

Some updates may require the notifier to be atomic i.e. either everybody is notified or nobody is! This could be particularly important for share price updates etc.

2.1.7 Accidental disconnection by customer

Note: This probably doesn’t belong here — it is more of a failure scenario, for the moment it is going here as a place holder.

Jim is reading a report he finds it incredibly boring and switches his machine off.

One way of getting around this problem is to have the browser cache all of the informationUnit. Caching and charging is atomic: it either caches it completely and register’s charges, or does neither. This means that Jim’s potential to corrupt the informationUnit is limited. The session information needs to persist and be dependable. This means that we probably cannot rely on Jim’s machine to store it.

At present this scenario concentrates on a text-based system. If audio and video is introduced the charging problems become much harder, because you are unlikely to be able to cache the whole video on all machines that want to be able to access it.

2.1.8 Discussion

The above scenarios concentrate on the interaction between the charger, browser and session. The notification service, installation and maintaining of

the informationUnits needs more thought. This probably means doing another analysis iteration.

2.2 Event flow diagram

Figure 2.7 shows an event flow diagram for the interaction between the customer, browser, session, informationUnit and charger. This diagram does not show a CopiedUnit, this is because interaction with a copiedUnit is subsumed by interaction with an informationUnit. (A copiedUnit supports a subset of the functionality of an informationUnit.)

Figure 2.7: Event flow diagram for interaction involving Customer

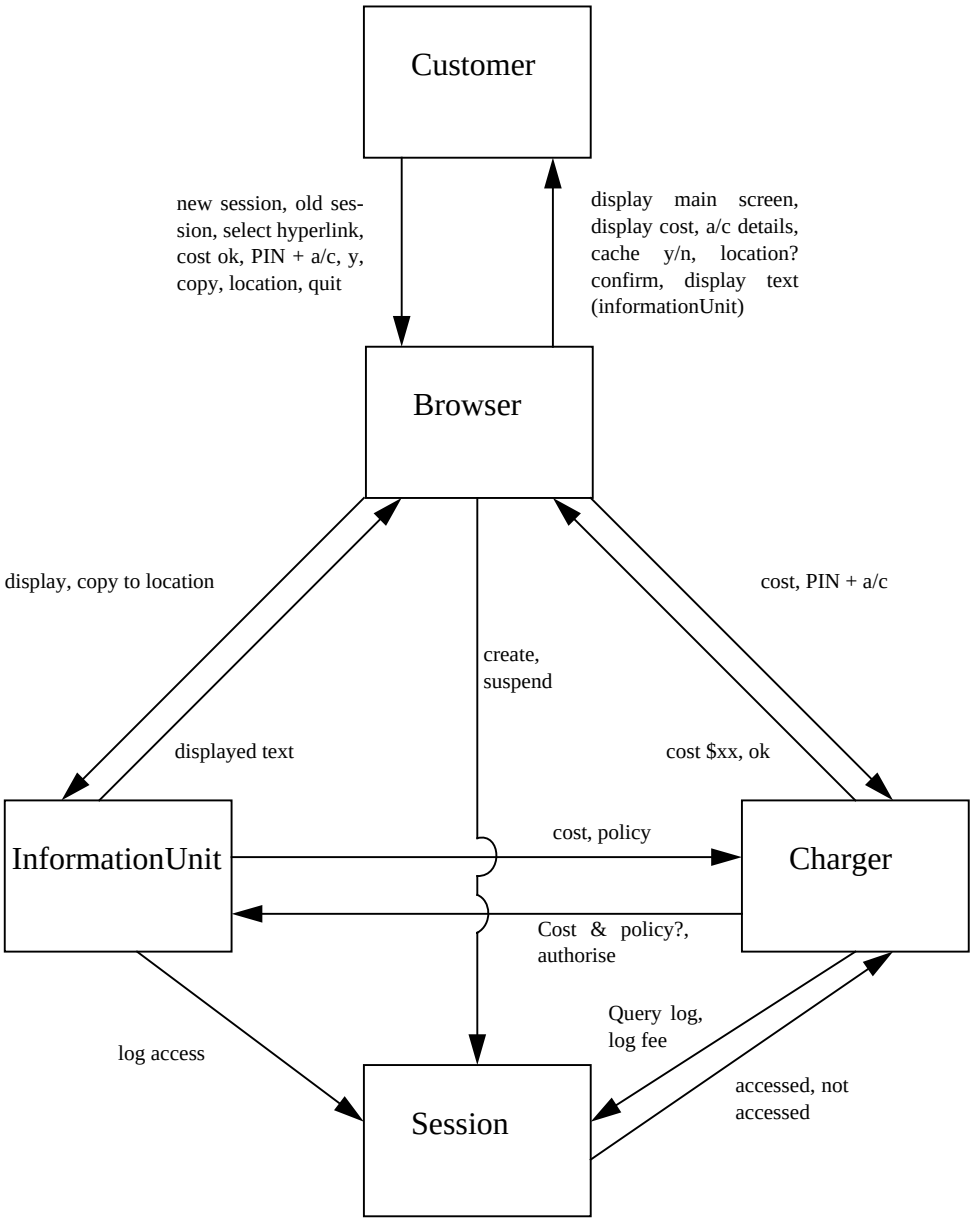
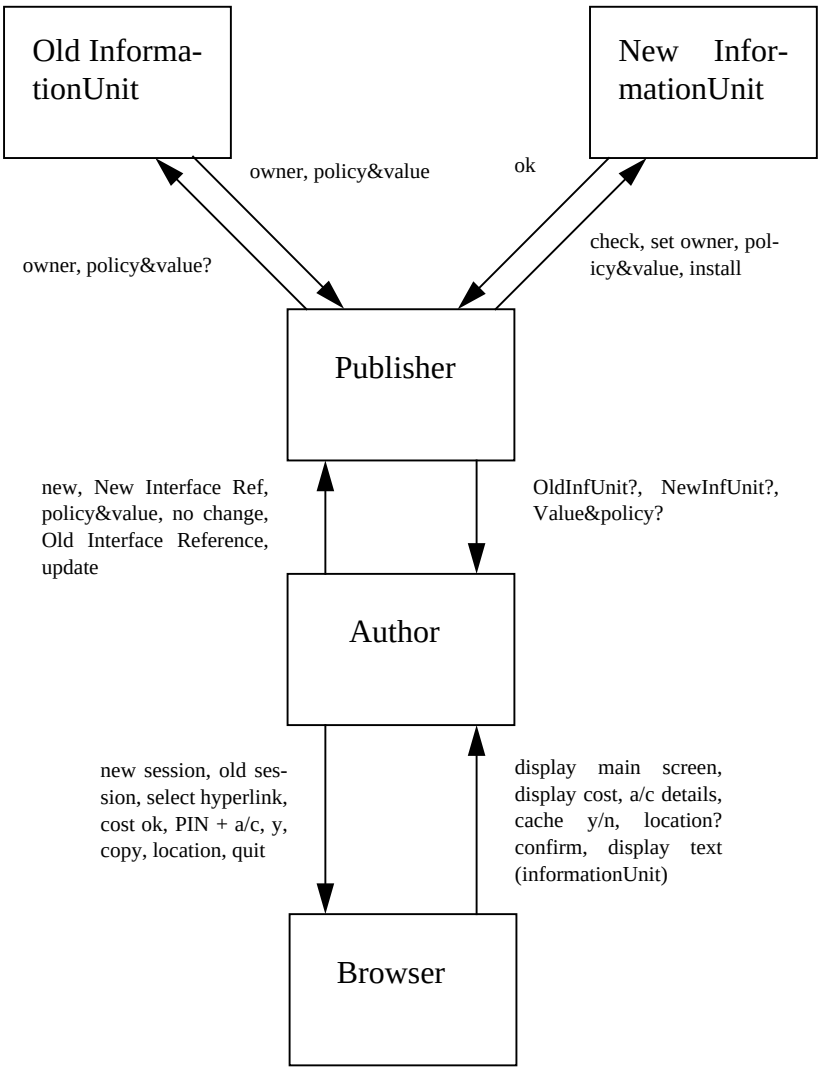


Figure 2.8 shows the event flow diagram for interactions involving authors.

Figure 2.8: Event flow diagram for interaction involving author



2.3 State Diagrams

From the previous sections it will be seen that the most complicated objects (at least at the level of abstraction being used here) are the browser, the InformationUnit, Charger and Session. This section presents state diagrams for each of these objects.

Figure 2.9: State diagram for Browser

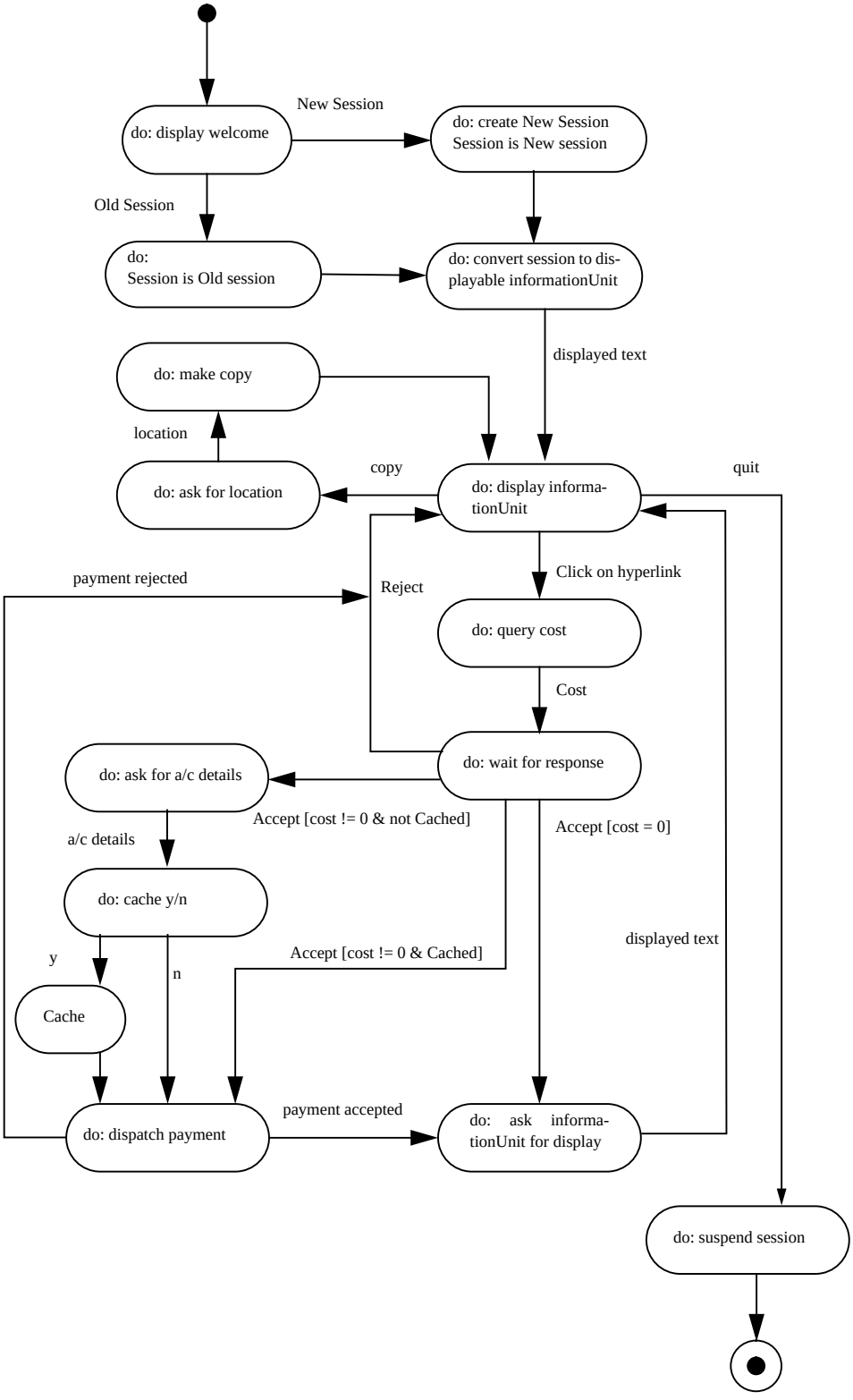


Figure 2.10: State diagram for InformationUnit

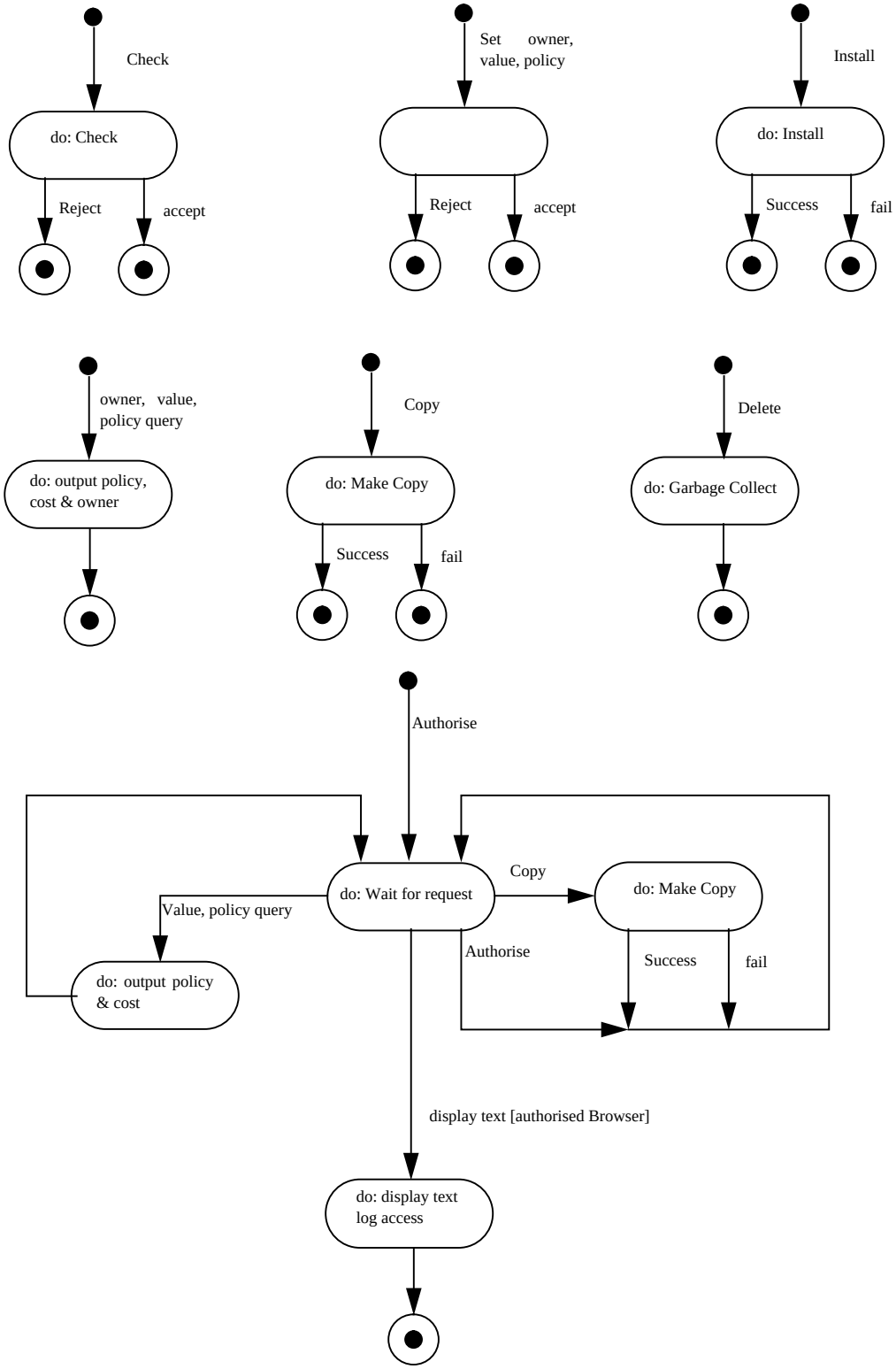


Figure 2.11: State diagram for Charger

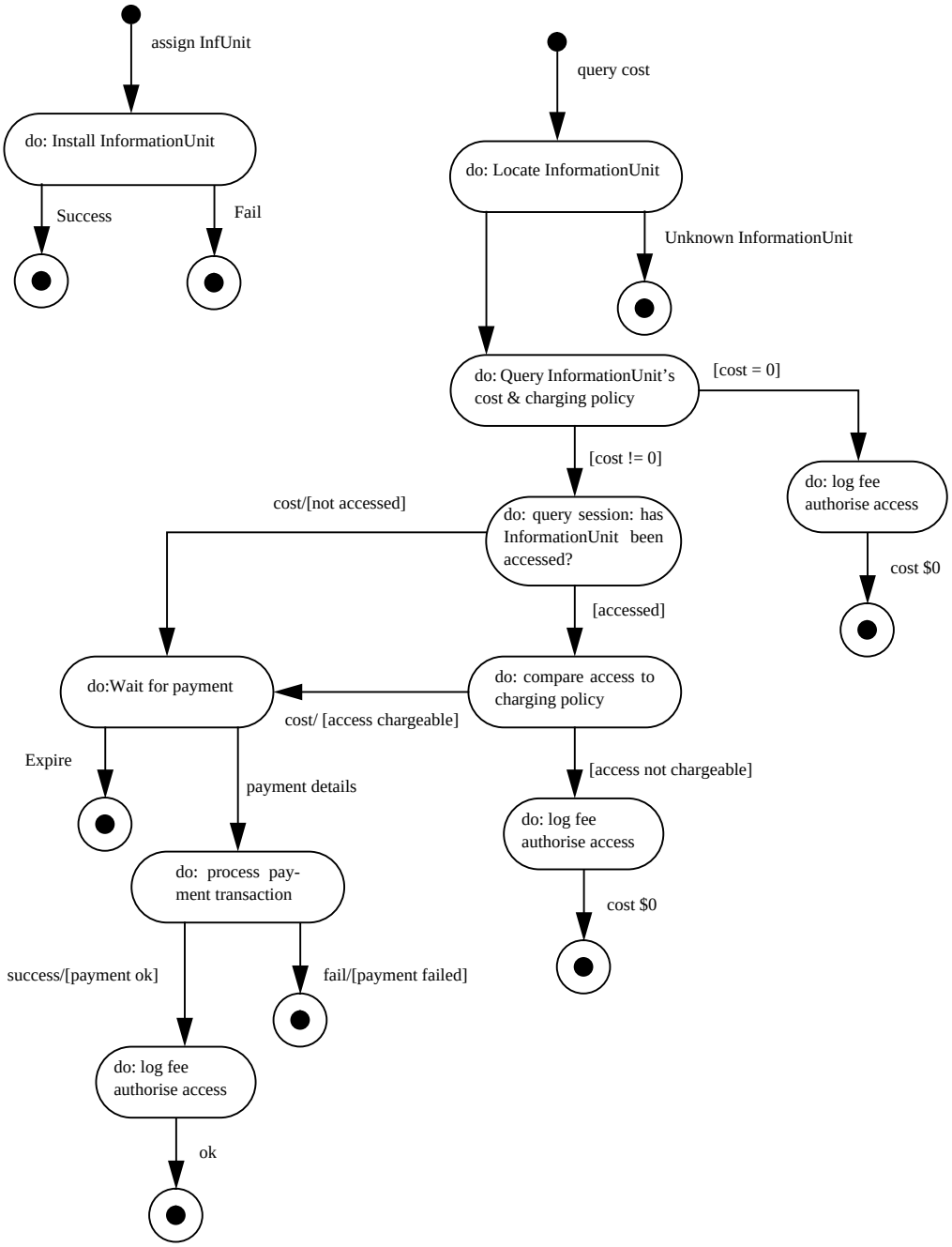
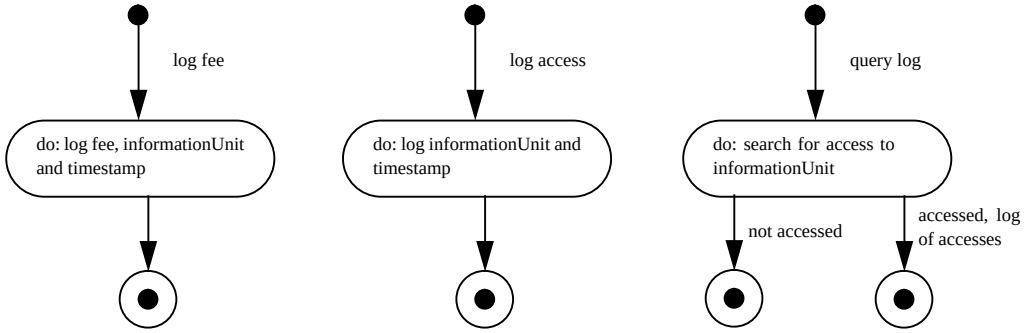


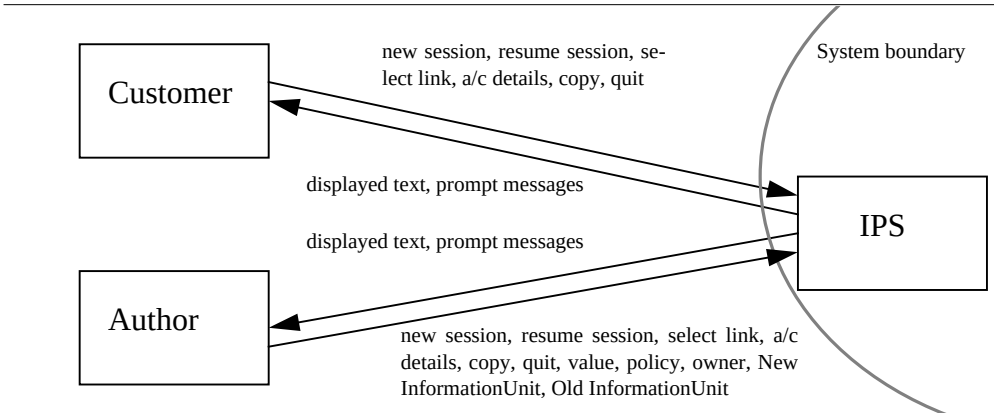
Figure 2.12: State diagram for Session



3 Functional model

This chapter presents a functional model of IPS using the methodology described in [RUMBAUGH 91]. Since IPS is driven by the activities of customers and authors, the functional model is less complicated and less important than either the object model or the dynamic model. Figure 3.1 shows the input and output data flows for IPS.

Figure 3.1: Input and output data flows



The top level data flow diagram is shown in figure 3.2.

Figure 3.2: Top level data flow diagram for IPS

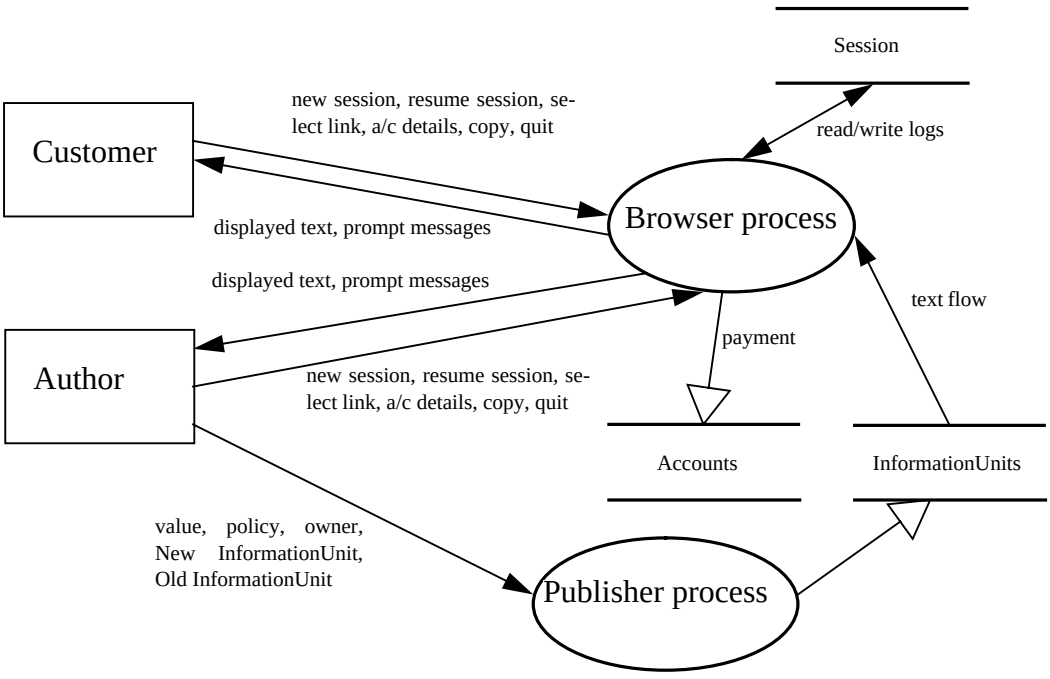


Figure 3.3 expands the browser process (quit and copy are not shown, because they are trivial (at least at this level of abstraction).

Figure 3.3: Data flow diagram for Browser

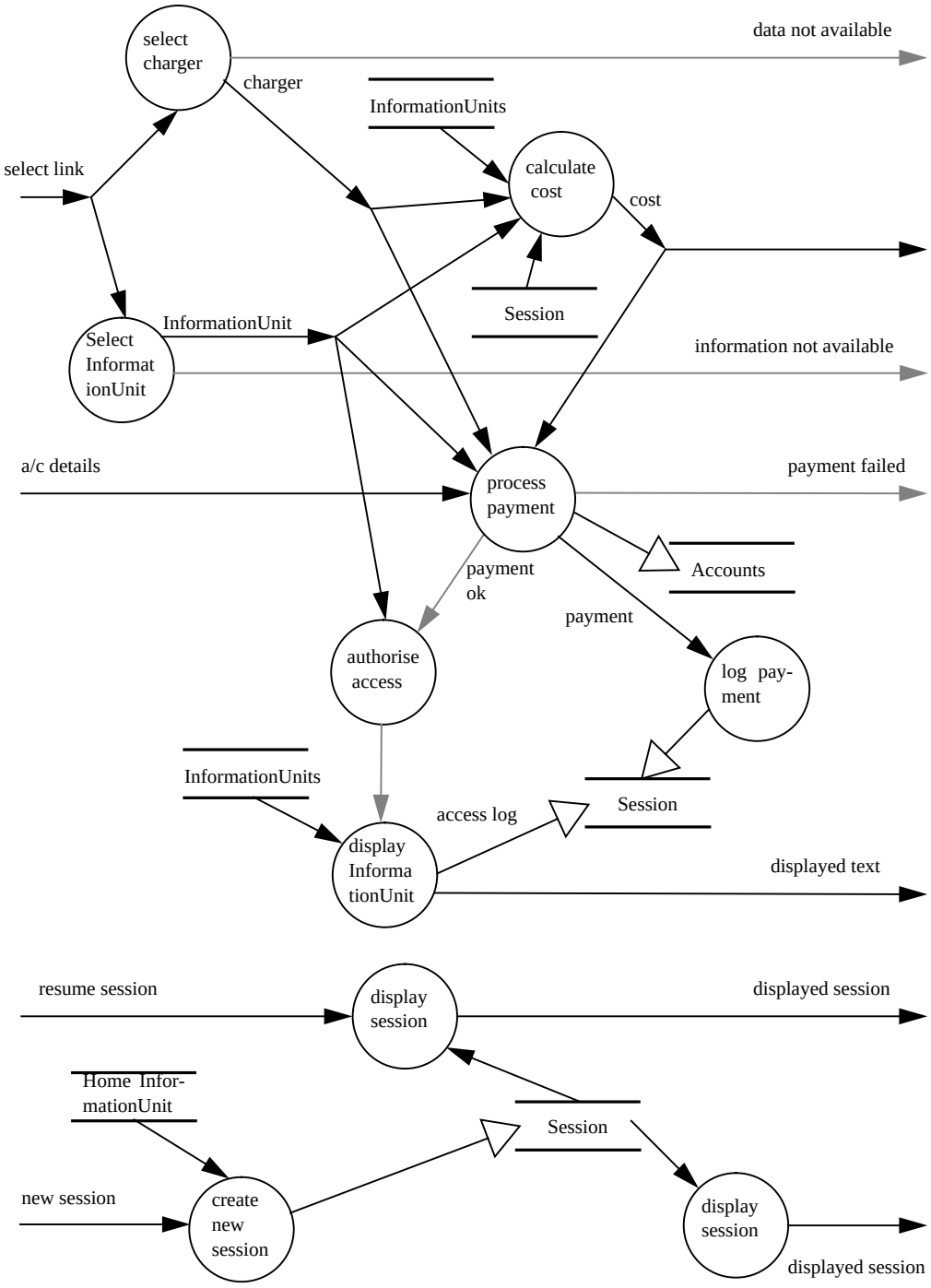
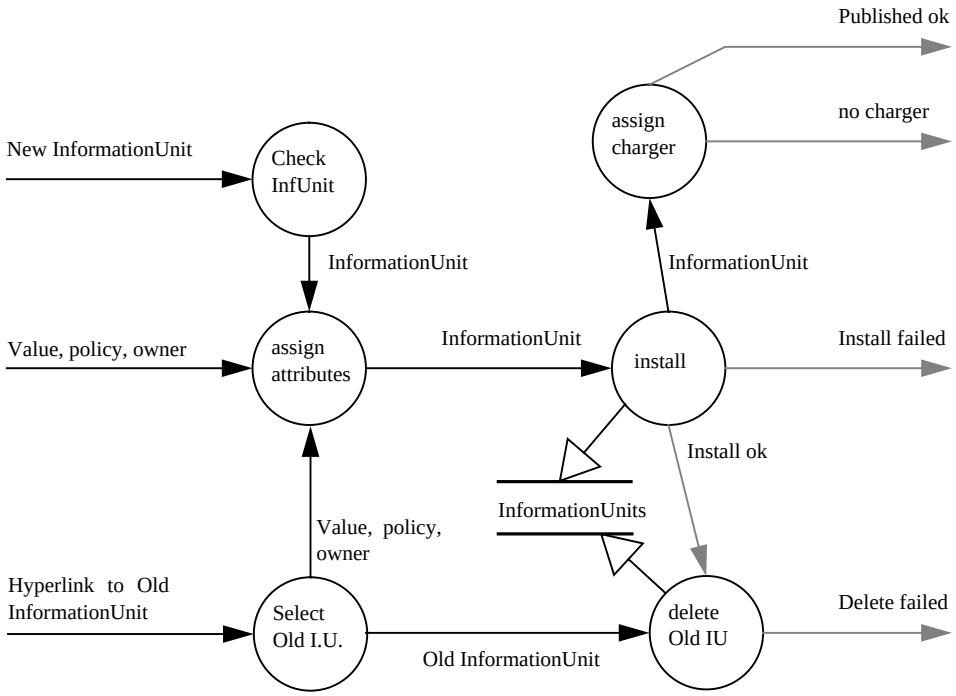


Figure 3.4 shows the data flow diagram for the publisher process.

Figure 3.4: Data flow diagram for Publisher



4 Future work

The analysis in this document identifies three major subsytems: the browser and session, the publisher and the notifier. An in-depth analysis is made of the browser, but only a sketchy analysis is made of the publisher and notifier. These latter two subsystems will doubtless introduce further dependability issues. Further investigation is required.

The system scenarios of chapter 2 deal with non-failure situations. We need to add scenarios which are failure situations (e.g. the customer's machine being switched off). We could use the failure model to perform an expectation analysis much like the responsibility analysis of chapter1. Once we have the failure scenarios we can begin to understand the options we have to make IPS dependable. The following section lists a number of possible technology options.

4.1 Technology Issues

Integrity of data is very important in this application. There is some inherent redundancy in the design which we may be able to exploit to try and protect integrity. The session objects record what a customer has accessed; this information could be used to audit the billing data.

We want to be able to ensure that no single failure of a site, or any entity can cause a service to become unavailable to customers. Relevant technologies include: replication, checkpointing and replicated stable stores.

A useful design principle is to minimise the state held be the service providers, the session objects facilitate this. This reduces the size and complexity of the servers and naturally distributes state which is likely to make it more available.

It may be a good idea to introduce monitoring processes to do continual on-line auditing. Organisations will want this information anyway so that they can keep usage statistics for business planning etc.

Another technology which might be applicable is the introduction of watchdog processes to monitor and audit the completion of service requests etc.

It is important to realise that we have little control over what customers might do. They may fail frequently: e.g. switch machine off because they get bored, they may have a mobile computer and disappear into a tunnel. This means we probably don't want customers to be responsible for committing transactions etc. Such critical functions should be performed on reliable nodes.

The converse of this is that we may want to allow customers to do some operations "disconnected", e.g. have access to local informationUnits. The billing data could be stored, waiting for a connection before it is forwarded. Access would be disabled once customers have reached their credit limit.

It may be useful to distinguish between logs or checkpoints that are kept for the recovery of individual objects and end-to-end logs which can be used to ensure that recovery of service users and providers is consistent.

We shall assume that settlement of bills etc. is outside of the scope of the system.

4.2 Acknowledgement

The authors would like to acknowledge the contribution of the following members of the ANSA team: Rob van der Linden, of APM Ltd.; Gomer Thomas, of Bellcore; Francis Wai, of APM Ltd.

References

[MORI 90]

Mori, R., Kawahara, M., “Superdistribution: The Concept and the Architecture”, Transactions of the IEICE; Vol. E-73#7 July 1990; Special Issue on Cryptography and Information Security.

[RUMBAUGH 91]

Rumbaugh J., Blaha, M., Premerlani, W., Eddy, F., Lorenson, W., “Object-Oriented Modeling and Design”, Prentice-Hall International 1991.

[SHARBLE 92]

Sharble R.C., Cohen, S.S.,“The Object-Oriented Brewery: A Comparison of Two Object-Oriented Development Methods”, Document BCS-G4059, Boeing, October 19th 1992.

[SHILLINGFORD 93]

Shillingford, J., “Online Databases — Windows adds appeal”, Financial Times, November 8th 1993.

Appendix A Further system scenarios

A.1 System Scenarios

A.1.1 Accessing an informationUnit within an existing session

The customer notices that Scoop's report mentions that Bogus Enterprises Inc. have announced the development of 5D television and points to an item published by NoNews describing the company's claims for the product. For brevity details of the interaction between the charger and session are omitted they would be the same as in §2.1.1.

1. The customer clicks on the hyperlink to the NowNews article
2. The browser obtains the reference to the report's interface and its charger interface.
3. The browser accesses the charger it finds that it will cost $1.00 to read the report.
4. The customer decides that the charge is ok and tells the browser to display the report.
5. The browser passes the cached account number an pin to the charger.
6. The charger verifies the details and debits $1.00 from the customer account.
7. The charger then tells the informationUnit that the appropriate fee has been paid and logs this information into the customer's session.
8. The browser asks the informationUnit to display itself and receives a stream of text.

A.1.2 Resuming a session and accessing an informationUnit

1. Jess starts up a browser and wants to resume her last session

Note: Does allowing people to have more than one session add any dependability issues? For the moment I cannot see how.

2. The browser asks her whether she wants to start a new session or resume the last session.
3. She clicks on the button marked last session.
4. The browser presents her with a representation of her last session. This is a graphical web-like representation of every informationUnit she has visited since she created a new session. Hyperlinks to the informationUnits are at the nodes of the web. This web can be treated exactly like any other informationUnit: by clicking on a hyperlink you access it.
5. She notices that several nodes in the web are highlighted indicating that the informationUnit has been updated since she last looked at it.

6. She clicks on one of the higlighted nodes.
7. The browser obtains the interface reference to the informationUnit and charger and goes through the sequence of actions described in §2.1.1 starting at step 11.
8. Jess finishes reading the updated report.
9. The browser returns her to the representation of her session.
10. She decides she would like to read “Probable and Possible Directions in Home Entertainment” from Scoop again, in the light of what she has just read.
11. She clicks on the node representing this report.
12. The browser obtains the interface reference to the report and its charger and then goes through the sequence of actions described in §A.1.1 starting at step 2.

In this case it means that Jess will not be charged for accessing the information. Whether or not this should be the case will be determined by the charger using the informationUnit's charging policy and the session to see what Jess has previously paid for and when.

Note: There are a number of dependability issues here: the session needs to accurately record what Jess has paid for and when, it also needs to be protected from tampering.

A.1.3 Accessing a copiedUnit

Jim is browsing a internal bulletin board dealing with developments in consumer electronics. He reads a note discussing the home entertainment market which has a link to a local copy of “Probable and Possible Directions in Home Entertainment”.

1. He clicks on the hyperlink.
2. The browser obtains the reference to the report's interface and its charger interface.
3. The browser accesses the charger and finds that it will cost $10.00 to read the informationUnit. (The charger queries Jim's session and the informationUnit to find this out see figure 2.1.)
4. Jim decides that the charge is ok and tells the browser to display the report.
5. The browser asks Jim for the payment method and is told to charge it to an account.
6. The browser asks Jim for a pin number and account name.
7. The browser asks Jim if the pin and account name should be cached, so that it will not be requested again in this session. Jim says yes.
8. The browser passes this information to the charger along with a reference to the session.
9. The charger verifies the details and debits $10.00 from Jim account.
10. The charger then tells the informationUnit that the appropriate fee has been paid and logs this information into Jim's session.
11. The browser asks the informationUnit to display itself and receives a stream of text.

Note at this point we have not said whether the charger is living at or near the CopiedUnit or if it lives in the originators system. At some point registering charges will involve accessing the originator's system. The failure scenario's will have to consider the need for a disconnect operation in which it is not possible to register charges.