A FORMAL BASIS OF IMPLICIT GROUP MESSAGING
TECHNICAL REPORT 616

DANIEL CUTTING AND BJÖRN LANDFELDT
SCHOOL OF INFORMATION TECHNOLOGIES
THE UNIVERSITY OF SYDNEY

AARON QUIGLEY
SCHOOL OF COMPUTER SCIENCE & INFORMATICS
UNIVERSITY COLLEGE DUBLIN

AUGUST, 2007
A formal basis of implicit group messaging

Daniel Cutting, Björn Landfeldt
School of Information Technologies
University of Sydney, Australia
{dcutting,bjornl}@it.usyd.edu.au

Aaron Quigley
School of Computer Science and Informatics
University College Dublin, Ireland
aquigley@ucd.ie

Abstract

This report both extends and provides a formal basis for prior preliminary work [5] on a novel messaging paradigm called implicit group messaging.

1 Introduction

Implicit group messaging (IGM) is a many-to-many messaging paradigm for delivering messages from publishers to groups of consumers. Publishers select groups of consumers by their attributes rather than by name or address. Any participant is capable of publishing messages to any implicit groups at any time and publishers do not need to be members of the implicit groups to which they publish. The actual membership of the same implicit group may vary from message to message as participants join and leave the system or as their attributes change over time. Publishing to an implicit group is termed casting. The messages are also referred to as casts.

Each participant in an IGM system has certain associated attributes or aspects of their identity. These are codified with a descriptive modelling language which can be tailored specifically for the application domain. An implicit group is defined as a subset of all participants that are described by a target expression over the same language. An expression conceptually segments participants into those that are described by the expression and those that are not. An expression is said to select those participants it describes.

Figure 1 shows a simple example of three different implicit groups, where each participant has associated sporting and musical interests. Implicit groups may be selected by combining interests, e.g. “football” (thick blue dashed line), “samba” (thin green dashed line), and “football & samba” (solid black line). Groups are not pre-defined but determined on an ad hoc basis. No participant, for example, explicitly joins or subscribes to the group “football & samba”, but such a group can still be specified.

Figure 2 illustrates a more complex example involving thousands of participants scattered over a wide area network. Implicit groups potentially cut across all network and geographic boundaries and may be extremely large or small (or even empty). Since implicit groups are not actually specified until publication, there is no trivial way to group participants in advance for the purposes of optimising network traffic for every potential group, in the way that IP multicast constructs minimal distribution trees, for example.

Implicit group messaging can be used as a fundamental messaging component in many domains, both social and technical. Collaborative group applications may use IGM to direct problems towards people that are likely to be able to solve them, or
to facilitate chat sessions between people with similar interests. Such applications may be appropriately combined with the concept of “communities of interest” but IGM is also applicable to machine-machine interaction and could be used to send commands to network nodes with particular configurations, or act as a resource discovery service.

Any concrete implementation of an IGM system should have the following properties:

1. All selected consumers eventually receive messages.
2. Messages are only delivered once to each consumer.
3. Only messages that are published are delivered to consumers.
4. Non-selected consumers do not receive messages.

Additional properties may also be desirable for some applications, such as the guarantee that messages are delivered to consumers in the same order they were published or that they will arrive within a certain amount of time. Such extensions to the IGM model are not addressed in this report. An implementation should also generally seek to minimise overall network and participant load, and deliver messages as quickly as possible, although the relative importance of these properties is application-specific.

When designing a model such as IGM without a directly applicable analogue, it is important to construct a theoretical foundation describing its salient features. This report continues by formally specifying IGM in Sections 2 and 3. Section 4 defines a specific “modelling language” for illustrative purposes, and inherent challenges associated with concrete implementations are then characterised in Section 5, particularly load balancing in the face of potentially large and spontaneously defined groups. Section 6 concludes the report.

2 Modelling implicit groups

The notation described in this section is summarised in Table 1. Implicit group messaging does not prescribe any particular modelling language to describe consumers and implicit groups; it may take many forms and can be chosen to suit a particular application domain. As such, the following generic definitions are independent of language, although Section 4 defines an exemplary concrete modelling language.

Let \( P \) be the set of all participants in the system. Each participant \( p \in P \) has a registration \( p_r \) which describes it according to a modelling language. In fact, a participant may have many registrations, but without loss of generality this report assumes exactly one registration per participant.

Any language used to model implicit groups must define a surjective function \( \triangleright \) (read “selects”) which maps a target expression and registration to \( \text{true} \) or \( \text{false} \). \( \mathcal{T} \) and \( \mathcal{R} \) are the set of all target expressions and registrations for a language respectively.

\[
\triangleright : \mathcal{T} \times \mathcal{R} \rightarrow \{\text{true, false}\} \tag{1}
\]

The notation \( t \triangleright r \) is read as “target expression \( t \) selects registration \( r \)”. \( \mathcal{P}_t \) is then defined as the implicit group (a subset of \( \mathcal{P} \)) selected by target expression \( t \).

\[
\mathcal{P}_t = \{ p \in \mathcal{P} | t \triangleright p_r \} \tag{2}
\]
3 Messaging implicit groups

This section formalises the properties of IGM enumerated above, based upon an interface of IGM operations, and specifies conditions that define which orderings of operations are legal. Any event-based messaging paradigm that supports distribution of messages from publishers to consumers can be specified similarly, including publish/subscribe and IP multicast, Mühl’s specification of publish/subscribe message [8], founded on “traces” and “linear temporal logic”, is thus adopted for IGM. In the interests of clarity and brevity, the specification presented here lacks some of the rigour required of a complete specification; the reader is referred to Mühl for a more rigorous treatment of trace-based specifications.

A minimal IGM model in which participants may only register and cast messages is presented. Implicit group members are individually notified by the system after a cast.

A subtle semantic aspect of IGM is that specific membership of an implicit group is dependent on the actual time at which it is reified. For example, peers that join a system after a publisher casts a message but before it is delivered should conceivably be members of the implicit group. Implementations that invoke a distributed design are even more susceptible to an ambiguous group definition because the decision to notify consumers may be made independently by many peers.

This dilemma also arises in publish/subscribe systems (and, in fact, any event-based system). It may be addressed by adjusting the definition of an implicit group to be those consumers that have completed a registration at the time they are notified of a cast. Note that in distributed systems this definition permits the possibility that there is no single “instant” at which the consumers notified concurrently form an implicit group according to the ideal definition in Section 2. Such a utilitarian definition clarifies system implementation and, in practice, the semantic effect on publishers and consumers is minimal.

An IGM system can be thought of as a state machine, where invocations of the operations in Table 2 move it between states. Different combinations of operations will move the system to different states, and will often be dependent upon the existing state. A trace is an infinite sequence of states interleaved with operations.

\[
\text{trace} = \text{state}_1, \text{operation}_1, \text{state}_2, \text{operation}_2, \ldots
\]

Any ordering of operations and states can form a trace, but only a small subset of these are legal for a system observing the IGM properties. For example, it is legal for the system to notify \(p\) of cast \(c\) (with target expression \(c_t\)) if and only if \(p\) has registered a selected registration such that \(c_t \sqsubseteq p_r\). Note that the states themselves can be omitted from the trace since it is the relative orderings of operations that are of interest.

\[
\text{trace} = \text{reg}(p), \text{cast}(q,c), \text{notify}(p,c), \ldots
\]

Trace 4 satisfies all informal properties of an IGM system. It would however constitute an illegal ordering if \(c_t \not\sqsubseteq p_r\) since \(p\) would be notified of a cast for which it had not previously registered a selected registration, violating Property 4 (Section 1).

The complete set of legal traces can be defined using linear temporal logic (LTL). This is a modal logic

<table>
<thead>
<tr>
<th>Table 1: Notation.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>IGM modelling languages</strong></td>
</tr>
<tr>
<td>(\mathcal{P}) Set of all participants.</td>
</tr>
<tr>
<td>(p_r) The registration of participant (p).</td>
</tr>
<tr>
<td>(c_t) The target expression of cast (c).</td>
</tr>
<tr>
<td>(\mathcal{P}_t) Implicit group selected by (t).</td>
</tr>
<tr>
<td>(</td>
</tr>
<tr>
<td>(T) All target expressions in a language.</td>
</tr>
<tr>
<td>(\mathcal{R}) All registrations in a modelling language.</td>
</tr>
<tr>
<td>(t \sqsupset r) Target expression (t) selects registration (r).</td>
</tr>
<tr>
<td>(C(q)) Set of all casts published by participant (q).</td>
</tr>
<tr>
<td>(R) Set of all registered participants.</td>
</tr>
</tbody>
</table>

Tag-based modelling language (Section 4)

| \(T(t)\) Target set of expression \(t\). |
| \(R(r)\) Registration set of registration \(r\). |
Table 2: The IGM interface.

<table>
<thead>
<tr>
<th>Method</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>reg(p)</td>
<td>Consumer p registers p.r.</td>
</tr>
<tr>
<td>cast(p, c)</td>
<td>Publisher p casts c to group selected by target expression c.t.</td>
</tr>
<tr>
<td>notify(p, c)</td>
<td>Consumer p notified of cast c.</td>
</tr>
</tbody>
</table>

that allows reasoning about systems along a future timeline. A set of operators allow statements to be made about which conditions can or must occur at future points in the trace. Three temporal operators are needed to specify the IGM model.

- □ X means X is true in all future states.
- ◯ X means X is true in at least one future state.
- ◦ X means X is true in the next state.

Using these operators, it is possible to define predicates that hold for some traces. For example, the predicate $\text{cast}(c) \Rightarrow \diamond \text{notify}(q, c)$ holds for all traces where q is notified of c immediately after c is cast.

$$trace = \text{cast}(c), \text{notify}(q, c), \ldots$$ (5)

This predicate may be weakened to allow notification at some point in the future rather than immediately: $\text{cast}(c) \Rightarrow \diamond \text{notify}(q, c)$.

$$trace = \text{cast}(c), \ldots, \text{notify}(q, c), \ldots$$ (6)

LTL is often used in conjunction with safety and liveness conditions. For IGM, these can be used to describe traces where unwanted operation orderings do not occur, and where desirable orderings eventually occur, respectively. A correct system satisfies both safety and liveness conditions. Such approaches have been used to specify and verify the correctness of other event-based messaging systems such as Rebeca [8] and Gryphon [3].

### Liveness

A liveness condition (Theorem 7 in Table 3) is used to express IGM Property 1. It states that if a consumer registers, it will eventually (and subsequently) be notified of all casts from any publisher that select its registration.

### Safety

A safety condition (Theorem 8 in Table 3) is used to express Properties 2, 3 and 4. It states that when a consumer is notified of a cast, it will not be notified again, the message was previously cast by some publisher, and its registration is selected by the cast’s target expression. Additionally, the condition requires that the consumer has completed a registration in order to agree with the adjusted definition of an implicit group. Let $C(q)$ be the set of all messages cast by q so far, and R be the set of all peers that have registered so far.

#### 4 A tag modelling language

An example of a concrete modelling language is provided in this section. It must be clear so as not to obscure IGM itself, yet sufficiently rich for a range of applications. It is based around the idea of simple descriptive tagging, although instead of tagging content as is frequently the case with Internet-based content, participants tag themselves. Target expressions used by publishers are strings that combine tags using conjunctive (\&) and disjunctive (\|) operators, and parentheses to modify evaluation order, according to the following grammar:

- $tag := [a - z]+$ (9)
- $expression := factor \ ("\| factor \ \| \& factor\)\ast$ (10)
- $factor := tag \ \| \("expression \)\)$ (11)

The selection function $\triangleright$ operates as expected under Boolean logic. For example, the target expression $(samba|jazz)\&football$ selects exactly those consumers that have registered the tags samba and football, and/or jazz and football.
Table 3: IGM Liveness and Safety conditions, respectively.

\[ \square \text{reg}(p) \Rightarrow \Diamond \square[(\text{cast}(q,c) \land c_t \triangleright p_r) \Rightarrow \Diamond \text{notify}(p,c)] \] (7)

\[ \square \text{notify}(p,c) \Rightarrow \Diamond \neg \text{notify}(p,c) \land \exists q,c \in C(q) \land c_t \triangleright p_r \land p \in R \] (8)

More formally, let a registration \( r \) describing a participant’s attributes be represented by a registration set \( R(r) \) containing a tag for each attribute. E.g., a registration for a peer with tags \textit{samba} and \textit{football} has a registration set \{\textit{samba}, \textit{football}\}. Let a target expression \( t \) be represented by a target set \( T(t) \) which is a set of target elements, each of which represents a disjunctive term of the expression. Each target element is a set of conjunctive tags. E.g., the expression \textit{(samba|jazz)}&\textit{football} expands to two disjunctive terms, \{(\textit{samba}&\textit{football}), (\textit{jazz}&\textit{football})\}, each containing two conjunctive terms. This expression is therefore represented by a target set \{\{\textit{samba}, \textit{football}\}, \{\textit{jazz}, \textit{football}\}\}. The selection function \( \triangleright \) can then be defined as \textit{true} if and only if any non-empty subset of the registration set is an element of the target set:

\[ t \triangleright r \Leftrightarrow \exists K \in T(t); K \subseteq R(r), K \neq \emptyset \] (12)

The definition of an implicit group (Theorem 2) can thus be restated for the tag-based modelling language using the selection function of Theorem 12 to yield:

\[ \mathcal{P}_I = \{p \in \mathcal{P}|\exists K \in T(t); K \subseteq R(p_r), K \neq \emptyset\} \] (13)

That is, an implicit group is the set of participants that have registered all of the tags in at least one element of a target set.

5 Characterisation of IGM

This section describes the types of peer registrations and cast distributions expected of IGM systems and discusses how these distributions may affect IGM implementations.

There are two types of data in a tag-based IGM system: the tags peers use to describe themselves; and the tags used in casts’ target expressions. These registrations and casts, like many complex systems, will likely follow Zipf’s law [9], which states that the frequency of a word in a language, \( F(w) \), is inversely proportional to its rank, \( R(w) \):

\[ F(w) \propto \frac{1}{R(w)} \] (14)

The law has been observed not only in large natural language corpora, but also in many Internet phenomena such as the number of visitors to web sites and the number of connections between autonomous computer networks like ISPs and universities [2]. Figure 3 shows two examples of actual Zipfian distributions on log-log scales: the frequency of all words appearing in the complete works of Shakespeare, and the top 500 categories assigned to articles on Wikipedia (on 17 June 2007) [1]. Zipf’s law has also been observed in the frequency distributions of tags used on recent social web sites such as the bookmarking site, del.icio.us [7, 6, 4].

Zipf’s law can be made more general by introducing an exponent \( s \) to the denominator, in order to describe its “skew”. An exponent of 0.0 describes a uniform frequency distribution (all elements are equally frequent), while exponents greater than 1.0 describe extremely skewed distributions where a few elements are vastly more popular than all others.

\[ F(w) \propto \frac{1}{R(w)^s} \] (15)

Although the precise exponents should be expected to vary across application domains, it can be expected that peer registrations and casts will follow roughly Zipfian distributions of varying skew.
Frequencies of words in the complete works of Shakespeare are approximately Zipfian.

Frequencies of top 500 categories assigned to Wikipedia articles.

Figure 3: Examples of real phenomena with Zipfian distributions.

Table 4: Various classes of implicit groups.

<table>
<thead>
<tr>
<th></th>
<th>Frequently cast</th>
<th>Seldom cast</th>
</tr>
</thead>
<tbody>
<tr>
<td>Common tags</td>
<td>large, frequent</td>
<td>large, seldom</td>
</tr>
<tr>
<td>Rare tags</td>
<td>small, frequent</td>
<td>small, seldom</td>
</tr>
</tbody>
</table>

Depending on the distribution of the tags registered by peers, variously sized implicit groups can result, ranging from empty to significant fractions of all peers. Similarly, the frequency with which the same groups are selected will vary with the skew of the casts. For illustrative purposes, implicit groups can be described as four classes caused by the distributions of peer registrations and casts (Table 4), although the actual distribution of group sizes and selection frequencies is not discrete.

When peer registrations are highly skewed, many peers register the same tags. Thus, target expressions composed of common tags typically define larger implicit groups than those using rare tags, and tend to cause greater load on an IGM system during the cast. Implicit groups may be potentially any size (i.e., they are $O(n)$, where $n$ is the number of participants in the system). If certain peers or servers in the system are responsible for handling individual casts, they will need to notify many consumers. Hence it is desirable to divide the responsibility for large implicit groups over many peers.

When the cast distribution is highly skewed, the same implicit groups are selected frequently. If particular peers are responsible for these casts, they will need to handle a high frequency of requests. Again, distributing this load over many peers is preferable. If the frequently selected groups are also large, an IGM system is placed under extreme pressure because many messages must be repeatedly delivered to the same large set of consumers. A fair and efficient IGM system must be capable of delivering casts to all of these classes of implicit groups without overloading peers either on a per cast basis or over many casts.

6 Conclusion

This report has formally specified implicit groups and implicit group messaging using trace-based specifications and safety and liveness conditions rooted in linear temporal logic. A concrete tag-based modelling language has also been defined for illustrative pur-
poses, and the types of expected implicit groups and attendant loading problems have also been described.

**Acknowledgements**

The authors would like to acknowledge the support of the Smart Internet Technology CRC.

**References**


