Cross-context Web Browser Communication with Unified Communication Models and Context Types

Ivan Zuzak, izuzak@gmail.com
School of Electrical Engineering and Computing, University of Zagreb, Zagreb, Croatia

Marko Ivankovic, ivankovic.42@gmail.com
Google GmbH., Zurich, Switzerland

Ivan Budiselic, ibudiselic@gmail.com
School of Electrical Engineering and Computing, University of Zagreb, Zagreb, Croatia
Agenda

• Motivation

  • Web Applications and Cross-Context Communication
  • Cross-Context Communication Ecosystem Systematization

• Pmrpc Library

  • Pmrpc API
  • Performance Evaluation

• Closing Remarks
Motivation

- Rich Internet Applications, Web 2.0
  - Mashups
  - Widget-based environments
  - Client-side background processing
Multi-context Web Applications

- Multi-context Web applications
Multi-context Web Applications

- Multi-context Web applications
  - Context types
    - Visual Window contexts = GUI + event loop (HTML+JS)
    - Background Worker contexts = event loop (JS only)
  - Cross-context communication
    - Origin-based context isolation
Multi-context Web Applications

- Multi-context Web applications
  - Context types
    - Visual Window contexts = GUI + event loop (HTML+JS)
    - Background Worker contexts = event loop (JS only)
  - Cross-context communication
    - Origin-based context isolation
- Similar to Operating Systems
  - Processes and inter-process communication
  - *The browser is the new OS*
Cross-Contex Communication Systems

- No research on analyzing properties of such systems
Cross-Context Communication Design Space

Communication initiation
Visual display of communication
Semantic interoperability

Latency
Authorization of sender and receiver
Communication integrity

Size
Authentication of sender and receiver
Communication confidentiality

Worker context support
Window context support
Naming
Cross-origin support
Type of system

Reliability
Distribution scheme
Discovery
Cross-application support
Communication model

User experience and high-level concerns
Performance
Security

Basic communication concerns (Programming level)
Cross-Contex Communication Ecosystem Systematization Results

- Window: 29 systems
- Worker: 6 systems
- Window and worker: 4 systems

Window/worker context support
Cross-Contex Communication Ecosystem Systematization Results

![Bar chart showing the number of systems for different communication models: Msg (17), Sh. mem (8), RPC (8), Pubsub (6).]
Cross-Context Communication Ecosystem Systematization Results

![Graph 1: Authorization of sender and receiver](image1)

- Authorization of sender: No = 23, Single = 3, ACL = 5
- Authorization of receiver: No = 26, Single = 3, ACL = 2

![Graph 2: Discovery and reliability support](image2)

- Discovery: No = 29, Yes = 2
- Reliability: No = 24, Yes = 7
Pmrpc Library

- Pmrpc
  - Cross-context communication JavaScript library
    - Advanced communication features
  - Rule of least liability
    - “Systems should minimize the liability that the user undertakes to ensure application security.”
    - Hiding complexity of cross-context communication is desired
Pmrpc Library

• **Client-side framework**
  • No server components used
  • Based on HTML5 and WebWorker postMessage primitives (secure message-passing mechanisms)
Pmrpc Library

- **Client-side framework**
  - No server components used
  - Based on HTML5 and WebWorker postMessage primitives (secure message-passing mechanisms)
- **Unified Web Worker and Window context support**
  - Wrap and unify browser primitives
Pmrpc Library

- **Client-side framework**
  - No server components used
  - Based on HTML5 and WebWorker postMessage primitives (secure message-passing mechanisms)
- **Unified Web Worker and Window context support**
  - Wrap and unify browser primitives
- **Unified communication models**
  - Message-based communication
  - Remote procedure call
  - Publish-subscribe
Pmrpc Library

- **Client-side framework**
  - No server components used
  - Based on HTML5 and WebWorker postMessage primitives (secure message-passing mechanisms)

- **Unified Web Worker and Window context support**
  - Wrap and unify browser primitives

- **Unified communication models**
  - Message-based communication
  - Remote procedure call
  - Publish-subscribe

- **Discovery support**
  - Discovery of contexts, procedures and channels
Pmrpc Library

• Client-side framework
  • No server components used
  • Based on HTML5 and WebWorker postMessage primitives (secure message-passing mechanisms)

• Unified Web Worker and Window context support
  • Wrap and unify browser primitives

• Unified communication models
  • Message-based communication
  • Remote procedure call
  • Publish-subscribe

• Discovery support
  • Discovery of contexts, procedures and channels

• Reliability support
  • Message-level handshake and retry mechanism
Pmrpc Library

- **Client-side framework**
  - No server components used
  - Based on HTML5 and WebWorker postMessage primitives (secure message-passing mechanisms)

- **Unified Web Worker and Window context support**
  - Wrap and unify browser primitives

- **Unified communication models**
  - Message-based communication
  - Remote procedure call
  - Publish-subscribe

- **Discovery support**
  - Discovery of contexts, procedures and channels

- **Reliability support**
  - Message-level handshake and retry mechanism

- **Authorization support**
  - Whitelist access control list mechanism
Pmrpc Architecture

- Not enough time, see the paper
- Pmrpc = wrapper for different context types, communication models, reliability …
Pmrpc API

- pmrpc.register(handler, procedureName, ?acl)
  - Register procedure (for RPC)
  - Subscribe to channel (for pubsub)
  - ACL whitelist of authorization rights

- pmrpc.unregister(procedureName)
  - Unregister procedure (for RPC)
  - Unsubscribe from channel (for pubsub)
Pmrpc API

  - Invoke procedure (for RPC)
    - `destinationContext` for addressing
    - `onSuccess` and `onError` handlers
  - Publish to channel (for pubsub)
    - Automatic discovery of destination contexts
  - ACL whitelist of authorization rights
  - Messages are retried in case of errors
Pmrpc API

Sender widget code

http://wikipedia.org/widget

Receiver widget code

http://maps.google.com/widget

pmrpc.call("refreshMap", ["31.0", "45.0"], 5, returnValueHandler);

function returnValueHandler(result) {
  // use result
}

pmrpc.register(handler, "refreshMap", whitelist: ["http://wikipedia.org"]);

function handler(parameters) {
  // use parameters;
  return result;
}
Pmrpc API

---

**Sender widget code**

pmrpc.call(
"refreshMap", // procedure name
["31.0", "45.0"], // parameters
5, // retries
returnValueHandler
);

function returnValueHandler(result) {
// use result
}

**Receiver widget code**

pmrpc.register(
handler, "refreshMap",
whitelist : ["http://wikipedia.org"]
);

function handler(parameters) {
// use parameters;
return result;
}
Pmrpc API

Browser

Sender widget code

```
http://wikipedia.org/widget

pmrpc.call(
    "refreshMap",       // procedure name
    ["31.0", "45.0"],  // parameters
    5,                 // retries
    returnValueHandler
);

function returnValueHandler(result) {
    // use result
}
```

Receiver widget code

```
http://maps.google.com/widget

pmrpc.register(
    handler, "refreshMap",
    whitelist : ["http://wikipedia.org"
);

function handler(parameters) {
    // use parameters;
    return result;
}
```
Performance Evaluation

• What is the tradeoff of complexity?
  • Performance

• Experimental measurements
  • Data transfer (round trip)
    − Native postMessage primitive
    − Pmrpc library
  • Different message sizes
Performance Evaluation

- Pmrpc is 4 times slower than postMessage
- Still in milisecond range
- Expected results
  - Serialization
  - Reliability
Conclusion

- Cross-context communication
  - The foundation of future Web applications
- Pmrpc
  - Open-source and free browser library (MIT license)
  - Hides cross-context communication complexity
  - Performance analysis
    - ~4 times slower than native browser primitives
    - Still in milisecond range, fast enough
Questions?

Thank you!

Pmrpc library:
http://code.google.com/p/pmrpc/

Contact:
izuzak@gmail.com
ivankovic.42@gmail.com
ibudiselic@gmail.com