

# **The Free Internet Application Services Manifesto**

## **Based on Open Systems Integration Tools (OSIT)**

Starting with

Free Services: [ByName.net](http://ByName.net), [ByNumber.net](http://ByNumber.net)

Free Protocols: [LEAPForum.org](http://LEAPForum.org), [IETF.org](http://IETF.org) ,

and

Free Software: [MailMeAnywhere.org](http://MailMeAnywhere.org), [VoRDE.org](http://VoRDE.org)

Mohsen Banan

[public@mohsen.banan.1.byname.net](mailto:public@mohsen.banan.1.byname.net)

Version 1.4  
February 20, 2002

# Contents

<b>1</b>	<b>About A Vision for Free Internet Application Services</b>	<b>1</b>
1.1	Document Organization . . . . .	1
1.2	Work in Progress . . . . .	1
1.3	Placeholder for Notes . . . . .	1
<b>I</b>	<b>Philosophy</b>	<b>3</b>
<b>2</b>	<b>Commercial Dimension</b>	<b>5</b>
2.1	Characteristics of Neda's Subscriber Services . . . . .	5
2.1.0.1	Initial Subscriber Services: ByName.net & ByNumber.net . . . . .	8
2.1.1	Why Will We Succeed? . . . . .	10
<b>3</b>	<b>Nomenclature: Free Software and Open-Source Software</b>	<b>13</b>
<b>4</b>	<b>The Big Picture</b>	<b>15</b>
4.1	Introduction . . . . .	15
4.1.1	Who Needs This? . . . . .	15
4.1.2	The Big Picture . . . . .	15
4.1.3	Why ksh? . . . . .	15
4.2	What Does Free Mean? . . . . .	15
4.3	Free Protocols . . . . .	16
4.4	General . . . . .	16
4.5	Specific . . . . .	16
4.6	Model and Terminology . . . . .	16
4.7	Making Free Software Cheap . . . . .	16
4.8	Free Services Based on Free Software . . . . .	16
4.9	Glue: The Missing Piece . . . . .	17
4.10	Solutions Based On Policies . . . . .	17

4.11	The Complete Free Services Picture	17
4.11.1	User Environment – Free Subscriber Environemnt	17
4.11.2	Subscriber Environment Premise Equipment	17
4.11.3	Free Service Provider Environemnts	18
4.11.4	Service Provider Environment Subscriber Premise Equipment	18
4.12	Free Services Software Perspective	18
4.12.1	Subscriber Environment	18
4.12.2	Service Provider Environment	18
4.13	How tangibles Fit Together	18
4.13.1	EOE’s Place	18
4.13.2	OSMT’s Place	18
4.13.3	ByName’s Place	18
4.13.4	LEAP’s Place	18
4.13.5	MailMeAnywhere’s Place	18
4.13.5.1	Architecture	18
4.13.5.2	Server Side – Software for Service Providers	18
4.13.5.3	Linux	20
4.13.5.4	Solaris	20
4.13.5.5	Windows NT	20
4.13.5.6	Client Side – Software for Users & Subscribers	20
4.13.5.7	Linux	20
4.13.5.8	Solaris	20
4.13.5.9	Windows 95/98/NT	20
4.13.5.10	Palm Pilot	20
4.13.5.11	Windows CE	20
4.13.5.12	Services	20
4.13.6	The Role of Operation WhiteBerry	20
4.14	OSMT Overview	20
4.15	Central Modelling and Management	23
4.16	Site and Cluster Switching	23
4.17	Backup and Replication	23
4.17.1	OS	23
4.17.2	Rsync Update	23
4.17.3	Backup	23
4.17.4	CVS	23
4.18	Configuration	23

<b>II</b>	<b>Subscriber Services: ByName and ByNumber</b>	<b>25</b>
<b>5</b>	<b>The ByName Service Components</b>	<b>27</b>
5.1	Introduction . . . . .	27
5.1.1	Model and Terminology . . . . .	27
5.1.2	At A Glance . . . . .	30
5.2	Subscribers . . . . .	31
5.2.1	bynameSubscribers . . . . .	31
5.3	E-mail Accounts . . . . .	31
5.3.1	bynameNspMail . . . . .	31
5.3.2	bynameNspMailLists . . . . .	31
5.4	Web Pages . . . . .	31
5.4.1	bynameNspWebServers . . . . .	31
5.5	LEAP . . . . .	31
5.5.1	bynameNspLEAPFeatures . . . . .	31
5.5.2	bynameNspTo97EMSD . . . . .	31
<b>6</b>	<b>The ByNumber Service Components</b>	<b>33</b>
6.1	Subscribers . . . . .	33
6.1.1	bynumberNspSubscribers . . . . .	33
<b>7</b>	<b>ByName Web Service</b>	<b>35</b>
7.1	ByName Web Architecture . . . . .	35
7.1.1	What is ByName? . . . . .	35
7.1.2	ByName E-mail Model . . . . .	35
7.2	CVS Tree Checkout . . . . .	35
7.3	What is Jetspeed? . . . . .	35
7.4	Jetspeed Templates . . . . .	36
7.4.1	New Account Template . . . . .	36
7.4.2	Anonymous ByName Front Page . . . . .	36
7.4.3	Define Default Portal for New User . . . . .	36
7.4.4	Static HTML Place Holder . . . . .	36
7.4.5	E-Mail Confirmation and Verification . . . . .	37
7.4.6	Template for Each portlet in the Anonymous Front Page . . . . .	37
7.4.7	Configure the layout and navigation bars . . . . .	37
7.4.8	Portlet Registration . . . . .	38
7.4.8.1	Examples of Registry Entry . . . . .	38

7.5	ByName Functionality Specification . . . . .	38
7.5.1	Ideas and Bugs Report . . . . .	38
7.5.1.1	Ideas – Future Implementation . . . . .	38
7.5.1.2	Known Bugs . . . . .	39
7.5.2	State Table . . . . .	39
7.6	External Hooks . . . . .	40
7.6.1	OSMT - byNameNspJetspeed . . . . .	40
7.6.1.1	Create a New Account . . . . .	40
7.6.1.2	Delete an Existing Account . . . . .	42
7.7	Plugs for IMP . . . . .	42
<b>III</b>	<b>User Environment</b>	<b>43</b>
<b>8</b>	<b>Introduction</b>	<b>45</b>
<b>9</b>	<b>Emacs Office Environment (EOE)</b>	<b>47</b>
9.1	About EOE . . . . .	47
9.1.1	About The E-O-E Package . . . . .	47
9.1.2	Basic Office Services . . . . .	47
9.1.3	Emacs Office Environment . . . . .	47
9.2	Configuration . . . . .	48
9.2.1	Site Customization . . . . .	49
9.2.1.1	eoe-load . . . . .	49
9.2.1.2	eoe-require . . . . .	49
9.3	Overview . . . . .	49
9.4	Overview . . . . .	49
9.4.1	Categories of Services . . . . .	49
<b>10</b>	<b>Web-Based User Environment</b>	<b>51</b>
<b>11</b>	<b>Linux User Environment</b>	<b>53</b>
<b>12</b>	<b>Closed But Common User Environment</b>	<b>55</b>
<b>IV</b>	<b>Software Distribution Center: MailMeAnywhere</b>	<b>57</b>
<b>13</b>	<b>MailMeAnywhere Open Source Software and Documentation Distribution Center</b>	<b>59</b>

<b>V Commercial Support: Neda</b>	<b>61</b>
<b>A Neda Clusters</b>	<b>63</b>
A.1 Uncluster . . . . .	63
A.2 Office . . . . .	63
A.3 Island . . . . .	63
A.4 DMZ . . . . .	63
A.5 Payk . . . . .	64
A.6 Subscriber . . . . .	64
A.7 Test . . . . .	64
A.8 Public . . . . .	64



# List of Figures

2.1 Neda's Subscriber Services at a Glance . . . . .	6
4.1 MailMeAnywhere – Server Side Software . . . . .	19
4.2 MailMeAnywhere – Client Side Software . . . . .	21
4.3 Open Services Management and Administration Platform . . . . .	22
5.1 ByName Layer . . . . .	28



# List of Tables

# Chapter 1

## About A Vision for Free Internet Application Services

The **Open Systems Integration Tools**, or **OSIT**, is a set of tools on top of which various consistent policies can be implemented.

The purpose of *A Vision for Free Internet Application Services* is to describe the use and purpose of these tools. The Vision includes:

- An overview of OSIT, the underlying philosophy as well as the big picture.
- OSIT as the glue for ...
- An overview of the free subscriber services: ByName and ByNumber.
- Software Distribution Center
- Commercial Support

### 1.1 Document Organization

*A Vision for Free Internet Application Services* is organized as a series of independent articles. Each of these articles stands on its own, and can be read and understood independently of the others. Together, these articles provide a complete picture of the OSIT.

### 1.2 Work in Progress

*A Vision for Free Internet Application Services* is a work in progress, and its component articles are in various stages of completion. Some of the articles are complete and some are in the process of being further developed.

### 1.3 Placeholder for Notes

This section serves as a placeholder for notes.

- Wireless Emphasis, OSIT Goals and Scalability.
- OSIT never hand edit a file

**Part I**

**Philosophy**



## Chapter 2

# Commercial Dimension

### 2.1 Characteristics of Neda's Subscriber Services

The key characteristics of Neda's Subscriber Services are summarized in Figure 2.1. In the following sections we describe these characteristics in greater detail.

#### Largest Possible Audience

A very significant aspect of our Subscriber Services business is that it is targeted to the largest possible audience. As noted above, it is targeted to all demographics, all user devices, and all networks. It will include a globally-accessible telephone voice interface, which uses speech recognition, text-to-speech, and IVR in non-data (pure voice) environments on *all* phones. The Subscriber Services will be infinitely scalable to support unlimited audience expansion.

Throughout the following discussion, we draw a distinction between **users** of our Subscriber Services, and **subscribers**. Anyone is free to make use of Neda's Subscriber Services, without paying any fees, and without forming any kind of formal relationship with Neda. We refer to a person who uses our services on this generic basis as a **user**. A user need not provide Neda with any formal identification or personal information beyond that required to access the services, and there exists no fiducial relationship between the user and Neda beyond that of the customary Acceptable Use policy.

A **subscriber** is a person with whom there does exist some sort of fiducial relationship, implying certain obligations on the part of both Neda and the subscriber. On the part of the subscriber, these obligations may include the payment of a service fee, and the disclosure of the subscriber's personal identity. On Neda's side, these obligations will include the protection of the subscriber's information and privacy, and guarantees of the quality of service.

For example, a subscriber may provide Neda with his/her credit card information, and will have the right to specify that he/she receive no unsolicited advertizing. A user, on the other hand, will typically not provide Neda with credit card information, and Neda will be under no obligation to shield a user from advertizing.

A good example of an existing Subscriber Service based on the user-type relationship model, as we have defined it, is provided by Yahoo. A good example of an existing Subscriber Service based on the subscriber-type relationship model is AOL. Our Subscriber Services will accommodate both types of relationship and usage model.

- 
- **Largest possible audience**
    - All demographics (mobile professionals, soccer moms, teenagers, etc.)
    - All devices (phones, PDAs, laptops, desktops)
    - All networks (Internet, GSM, CDPD, iDEN, etc.)
    - Powerful, globally-accessible, telephone voice interface
    - Infinitely scalable
    - Fully exploits Internet & domain hierarchy
    - Highly distributable, designatable & franchisable
  - **Personal and customized virtual community**
    - Oriented towards interpersonal messaging: e-mail, voice-mail, Fax, etc.
    - Buddy list, chat, special interest groups, dating, multi-player games
    - News, stock quotes, weather, sports, traffic, airline information
    - White pages, yellow pages
    - E-commerce
  - **Making it widespread**
    - Make it free (supported by advertising, content provider, etc.)
    - Give away LEAP for all wireless data devices and environments
    - Leadership of LEAP will give us the attention we need
    - Emphasize, encourage and enable mobility, urgency and frequent usage
    - Build user base by paying carriers to bundle our services
  - **Completely open & free**
    - Based on truly open protocols
    - Implemented based on free and open-source software
    - Preference for, and encouragement of usage of, free client and device software by subscribers and users
  - **User oriented**
    - An agent of the subscriber
    - A guardian of the user's information
    - Full respect for the user's privacy
    - A facilitator – not the owner of information
    - A Buyer's Agent for the user – an Infomediary model
- 

Figure 2.1: Neda's Subscriber Services at a Glance

### **Personal and Customized Virtual Community**

Neda's Subscriber Services will take the form of a highly personal and ever-expanding virtual community. The services will be centered around interpersonal messaging using all appropriate media, including e-mail, voice-mail, Fax, etc. The services will provide users with a comprehensive suite of facilities and forums for interpersonal, communal, and social communications, such as: buddy lists, chat rooms, special interest groups, dating agencies, and multi-player games.

In addition, the Subscriber Services will provide users with various forms of information. This will include perishable information such as news, stock quotes, weather, sports, traffic, airline information etc., and non-perishable information such as white pages, yellow pages, dictionary lookup, etc.

### **Making it Widespread**

The Subscriber Services will include several characteristics designed to make their usage widespread. First, they will be provided to users free, and will be supported by advertising, content providers, etc.

Also, LEAP software implementations will be given away free, for all wireless data devices and environments.

Initially, our Subscriber Services will emphasize and focus on mobility, and the delivery of time-critical information. The opportunity to manage a user's mobility needs will provide us with frequent access to the user.

We will build a user base by paying carriers to bundle our services, and by sharing revenue with them. In addition, we will provide a cash and equity payment to carriers, providing them with a powerful incentive to open their networks.

One of the key ways in which we will expand our audience is by means of **franchise** operation. We view our Subscriber Services as a powerful core of computing and communications services which are accessed by users. There is nothing to prevent this entire set of Subscriber Services from being packaged and provided to an independent service provider, who then provides identical services to a particular demographic or subset of users.

For example, a third-world country might have only primitive and undeveloped Subscriber Services available to its consumers. An entrepreneur wishing to provide Neda's far superior Subscriber Services to this demographic could franchise the entire packaged set of Subscriber Services, then take responsibility for delivering them to this particular demographic. Note that the inherent hierarchy of the Internet and its domains provides an ideal structure for the creation and management of such franchises.

Clearly, this is an enormously powerful propagation and growth mechanism. Equally clearly, the financial rewards are enormous for the company that happens to be operating the franchise.

### **Completely Open & Free**

The Subscriber Services will be completely open and free; they will be based on open protocols, and implemented based on free software. In addition, the Subscriber Services will express preference for, and encourage usage of, free client and device software by subscribers and users.

This is an entirely new approach to the Subscriber Services business. The tremendous power of open protocols and free software has not been fully exploited by any other existing Subscriber Service. The primary beneficiary of the open-source software movement will be the Subscriber Service provider who understands and makes use of it to the fullest. Other service providers have a liability in terms of the assets that they have built.

### **User-Oriented**

The Subscriber Services will be oriented first and foremost to the needs and desires of the user, NOT those of the advertisers, content providers, or vendors. Nor will they be oriented to the benefit of Neda itself, the Subscriber

Services provider and virtual community organizer. What this means is that the services will remain as passive, silent, and anonymous as the subscriber wishes. For example, if the subscriber wishes to see no advertizing, he/she will see none. If the subscriber does not wish to be presented with unsolicited product and service offerings, then they will not be presented.

Because of our position as Subscriber Service providers and community organizers, during the process of building user and subscriber relationships, we will have access to user and subscriber usage information resulting from users' access activity and experiences. The gathering of such information is often referred to as "data mining," and is frequently used for targetted advertizing. However, we will only use this method and model when the user or subscriber explicitly requests or authorizes this. As an element of our user-oriented role, we will use this information when acting on the subscriber's behalf as a Buyer's Agent in the "Infomediary" model [?].

Only when the subscriber explicitly requests product, service, vendor, or any other form of commercial information will it be presented to him/her. The subscriber may request such information on a one-time basis, in which case the system will then return to its former unobtrusiveness until called upon again. Or, the subscriber may authorize well-defined and qualified information to be presented to him/her on an on-going basis.

In either case, the Subscriber Services will act in the best interests of the subscriber. The services will make full use of the subscriber's information for this purpose, but will act as a guardian of this information, and will fully respect the subscriber's privacy.

### 2.1.0.1 Initial Subscriber Services: **ByName.net** & **ByNumber.net**

Our initial, practical Subscriber Services implementation consists of two services: **ByName.net**, and **ByNumber.net**. As a starting point implementation, these initial services currently provide only a basic set of services to the mobile user. Eventually, the functionality of these services will be expanded to include the full range of capabilities listed in Figure 2.1.

As noted previously, some of the key characteristics of our Subscriber Services are that they are user-oriented, highly personalized, and show emphasis and respect for the user's identity above that of the service provider.

Our initial Subscriber Services implementation reflects all of these characteristics. The domain names **ByName.net** and **ByNumber.net** themselves reflect several of the characteristics of our Subscriber Services. First, they reflect their user-oriented nature. One of the aspects of our user-orientedness is that we are attentive to the needs of the user, but discreet – like a good butler. The names **ByName** and **ByNumber** reflect this self-effacing discretion: these names include no self-promotion of Neda whatsoever. Rather, the **ByName** and **ByNumber** services place primary emphasis on the user's identity; **ByName** is based on the user's name, while **ByNumber** is based on a numerical user ID.

The names **ByName** and **ByNumber** also reflect the very large intended scope of these services. Note that these names imply no particular type of service (e.g. wired, wireless, fixed, mobile); therefore they imply *all* types of service. The **ByName** and **ByNumber** services also emphasize mobility for the user.

**ByName** provides a set of free services, based on free protocols which have been implemented as free software. The **ByName** service provides each user with his/her own personal domain, using a naming convention based on the user's name. For example, a certain well-known television celebrity might be provided with the domain:

```
homer.simpson.1.ByName.net
```

**ByName** thus includes the user's own name as part of the domain. The user can then use this single domain for all his communications needs. Homer can administer, manage and control his personal communications through his personal domain; and through this domain, **ByName** will provide Homer with a comprehensive set of Mobile Messaging, e-mail, personal web, and other open-ended services. The demand for these sorts of personal services is becoming more and more evident as the users of first-generation and conventional service providers become increasingly sophisticated.

By appending various selectors in front of the @ sign, Homer can be provided with a number of separate addresses and mailboxes, such as `personal@homer.simpson.1.ByName.net`, or `office@homer.simpson.1.ByName.net`. Other prefix selectors which Homer can use are: `urgent`, `public`, `mobile`, `pager`, `fax` and `emergency`.

This provides our anti-hero with a consistent set of e-mail boxes that he can use for different purposes – one address for personal mail, a different one for work-related mail, and so on. Homer now has control over the routing of his e-mail without having to use a mail sorter or filters.

The user's home page is also based on his name; Homer's is

`http://homer.simpson.1.ByName.net`

All of the above is in sharp contrast to the way Subscriber Services are being provided today. A conventional service provider typically provides the user with a single e-mail address, usually of the form "SomeName@SomeDomain.com," where the name "SomeDomain" serves to identify and promote the Service Provider. This provides the user with a single mailbox, to which all mail for that address is sent.

This becomes inconvenient when the owner uses the account for multiple types of incoming e-mail. For example, the user may use the account for both personal and work-related mail, to subscribe to various mailing lists, and to participate in usenet groups. Over time the user may get onto a large number of mailing lists, resulting in an incoming e-mail stream spanning a very wide dynamic range of importance, from urgent personal e-mail, all the way down to meaningless spam.

E-mail applications typically deal with this by providing the user with tools to manage and prioritize mail. These consist of inbox sorters and filters to eliminate spam and prioritize incoming messages based on the originator or subject.

The ByName.net service provides a better way. ByName provides the user with multiple mailboxes and addresses, each of which can be dedicated to a particular type of e-mail. These various addresses have a simple and uniform naming scheme, based on the one symbol that is most personal to the user: his own name.

To learn more about the ByName service and to apply for an account, see the website at <http://www.ByName.net>.

For more information on accessing Neda Personal Computing and Communications Services, see the subscriber access manual [?].

The ByNumber.net service provides a complementary service to ByName, based on numbers rather than letters. ByNumber enables devices with digit-only origination capability (e.g. conventional telephone keypads) to send e-mail messages, and provides a unified way of sending messages to pagers, two-way pagers, faxes and e-mail accounts.

ByNumber also includes a telephone voice interface which provides speech recognition, text-to-speech, and IVR capabilities on *all* phones. In this way the full functionality of our Subscriber Services will be made available through ByNumber, just as they are through ByName. The ByNumber service is primarily accessible through an Interactive Voice Response (IVR) system using a standard touchtone telephone. At present, ByNumber only supports user origination of messages by way of:

- One-Way Paging (e.g., numeric pagers, alpha-numeric pagers)
- Mobile Messaging (e.g., Enhanced Two-Way Paging (ETWP))
- E-mail (e.g., Internet E-mail, AT&T PocketNet Mail)
- Fax

However, ByNumber.net is intended eventually to become a full companion to ByName.net, for voice access.

You can access the ByNumber Interactive Voice Response (IVR) system by calling 425-644-2972.

To learn more about the ByNumber service, see the website at <http://www.ByNumber.net>.

To learn more about how the ByNumber service works, see the *ByNumber User's Guide* [?].

ByName.net and ByNumber.net provide the starting point for our Subscriber Services. We will build on this starting point, and eventually endow our services with all the characteristics listed in Figure 2.1.

Other service providers are rushing to expand their user base immediately and as fast as possible. However, our business strategy does not call for us to create a user base during Phase I, and we have no plans to begin building one at this time. We will not begin to create and expand our user base until well into Phase II. Until that time, we will instead focus on creating a comprehensive, reliable and rapidly scalable Subscriber Services infrastructure. We are confident that when the time comes, the uniqueness and power of our Subscriber Services model will be a compelling motivation for their widespread adoption.

### 2.1.1 Why Will We Succeed?

These are necessary conditions for success; however, they are not sufficient. Even with all those conditions satisfied, two important questions remain. First, the Subscriber Services business is enormous – how can a small company like Neda succeed in capturing this business? Second, established Subscriber Services players such as AOL and Yahoo are already in existence. As newcomers, we clearly are at a disadvantage to these established companies. The fact that Subscriber Services is an inherently increasing returns business places latecomers at a further disadvantage. Why will Neda succeed as a small player in such a huge arena, a newcomer, and in competition with established providers such as AOL and Yahoo?

There are, in fact, three extremely good reasons why we will succeed in the face of these challenges. The first reason is that free and open protocols are an essential component of the Mobile Messaging industry. Any viable Subscriber Services solution must be protocol-based, and nobody besides Neda has the necessary efficient protocols for the mobile and wireless environment.

Without such protocols, all efforts by existing virtual communities to expand into this environment will fail, because they will be closed solutions, they will be in competition with one another, and they will further segment the industry. Other companies who pursue this same goal, regardless of how much money they invest, will only create increased industry fragmentation because of their lack of a common set of protocols. The consumer needs and wants what Neda alone is providing: protocols as a basis for convergence.

The second reason is that, because of our Three Business Units, Neda will be a provider of **all three** of the key components required to make the Mobile Messaging industry work: protocols, products, *and* Subscriber Services. The existing players only have the assets to provide the final component.

(Of course, AOL or Yahoo can readily acquire the missing assets, either from Neda, or from a company that chooses to compete with Neda on this basis. However, these companies will be quick to recognize the advantages of in-house ownership of these assets. And an astute businessperson will soon recognize that the quickest and cheapest way to acquire the missing assets is to buy out the company that has the greatest investment in these assets – in other words, Neda.

All that Neda has to do to make this a very real possibility is to become a credible threat to the existing Subscriber Services companies. For this reason, we view buy-out by a larger player is a very plausible and acceptable future scenario for Neda. And of course, this would provide a convenient and acceptable exit vehicle for our investors.)

Both of the above are compelling reasons for why we can compete and succeed in the Mobile Messaging arena. However, they say nothing about why we can succeed in the much broader Subscriber Services arena. For this we turn to our third reason for success: We represent the second generation of Subscriber Service providers, and can benefit from the experiences of our predecessors. Existing installed Subscriber Services evolved into their current model in an unplanned, ad hoc way. A Subscriber Services model designed from the ground up on the basis of the ad hoc blunders

and errors of the past five years will have a significant advantage. In particular, history has shown the enormous power and importance of the completely free and open Subscriber Services model. Neda's services will be based entirely on free protocols and free, open-source software. Not only will the software implementations of the LEAP protocols be open-source, but in fact *the entire Subscriber Services software architecture* will consist of free software.

A new entry into the Subscriber Services business requires at a minimum (a) a new and compelling value proposition for the user, and (b) a method for wide and large scale exposure. LEAP qualifies and provides an entry point on both counts, and also brings the power of open and free into the equation.

Furthermore, our Subscriber Services will be designed to be franchisable and distributed. We will franchise the service by moving computing and communications services closer to the subscriber, and we will gain the trust of the subscriber by putting him in control of the service. Under this model free software is an asset, whereas commercial software is a liability. Also, the current assets, investments and commitments of existing Subscriber Services become a liability under this model.

This represents a radically new way of looking at, developing and maintaining software, and it has profound consequences. What this means is that the entire Internet engineering and technical community who devote themselves to working on open and free protocols and software have in effect become an extension of our own research and development division. The work of this army of talented and committed people can be integrated into our Subscriber Services on an ongoing basis. Our unique, groundbreaking ability to harness the extraordinary power of openness and freedom sets us apart from any other service provider, and represents a further compelling argument for our success.

Furthermore, our Subscriber Services model is based on the Internet end-to-end model. This model places the user squarely in the driver's seat. The user may choose the best access device and software based on competitive features such as performance, price, or whatever criteria are of most importance to the user.

Our Subscriber Services are thus structured in a radically different way to those of AOL and Yahoo. They are structured in a way which is enormously powerful, and which those other companies cannot realistically be expected to emulate. It is on this basis that Neda can enter and dominate the Subscriber Services business.



## Chapter 3

# Nomenclature: Free Software and Open-Source Software

Throughout *A Vision for Free Internet Application Services*, we will make frequent use of the terms **free software**, and **open-source software**. These terms are often used informally and casually in the software industry, and may have several different meanings. For example, the term open-source software is often used to indicate merely that the source code is available. When we use the term open-source, however, we are using a much stricter definition than this.

Our definitions of the terms free software and open-source software are consistent with those of the Free Software Foundation and OpenSource.org. Throughout this business plan, we will use these terms with the following definitions:

**Open-source software** refers to software for which the source code is readily available. Either the software must be distributed as source code *per se*, or the source code must be readily available from some other source. Also, the software must be distributed under a license which does not restrict the selling or re-distribution of the software as a component of an aggregate software distribution. The license may not require any fee or royalty for such sale. Finally, the license must allow modification and the creation of derivative works, and must allow them to be distributed under the same terms as the original license.

The above describes only the essential, key characteristics of open-source software. For a complete and detailed definition, see <http://www.opensource.org/osd.html>.

**Free software** is software which the user is free to run, copy, distribute, study, change, and improve. For a complete and detailed definition, see <http://www.fsf.org/philosophy/free-sw.html>.

Note that in the term “free software,” the word “free” is being used to indicate the concept of *liberty* or *freedom of action*, rather than that of *without cost*. In other words, we mean free as in “free speech,” rather than “free beer.”



# Chapter 4

## The Big Picture

### 4.1 Introduction

#### 4.1.1 Who Needs This?

#### 4.1.2 The Big Picture

The free software movement is just the tip of the iceberg. Much remains to be done and to be seen. The next stage is the application of free software principles to create full and deep integrated computing and communications services.

The challenge is how to bring together large and independently developed pieces of software into complete, ready-to-use solutions.

The solution involves more than just software.

1. Free Software
2. Free Protocols
3. Free Services

#### 4.1.3 Why ksh?

ksh has evolved and matured as a result of extensive user feedback. It has been used by many thousands of people at AT&T since 1982, and at many other companies and universities. A survey conducted at one of the largest AT&T Bell Laboratories computer centers showed that 80% of their customers, both programmers and non-programmers, use ksh.

ksh is compatible with the Bourne shell. Virtually all programs written for the Bourne shell also run with ksh. If you are familiar with the Bourne shell, you can use ksh immediately, without retraining. The new version of ksh is compatible with earlier versions of ksh. ksh is readily available. ksh is extensible.

### 4.2 What Does Free Mean?

The English word “free” has two (at least) sets of meanings:

1. Having zero cost – i.e. free from financial obligation
2. Allowing unrestricted action – i.e. free from hindrance

### **4.3 Free Protocols**

### **4.4 General**

### **4.5 Specific**

1. OSMT
2. MailMeAnywhere (Software Distribution)
3. Free Services
4. Commercial Software and Services

### **4.6 Model and Terminology**

Inadequate and non-existence

### **4.7 Making Free Software Cheap**

The cost of ownership for free software is very high. Full integration of free software with free services is one of the ways of making free software cheap.

### **4.8 Free Services Based on Free Software**

Most of the use of free software so far has been on the server side. However, such use has not been in the context and towards the goal of creating a set of consistent services based on free software.

Adding a set of consistent policies and defining the goal of providing free services is next on the agenda.

Key attributes of free services:

- Based on Free Software
- Not controlled by software vendors (MS)
- Not controlled by the service provider. Not like AOL, MSN, etc.
- Including provision for governance by the User.
- ByName.Net is an initial example of Free Services.
- Highly distributed and franchised.

## 4.9 Glue: The Missing Piece

OSMT is the virtual glue that brings together large pieces of software and creates consistent free services and free user environments.

GNU/Linux is host-based and host-focused. OSMT is needed to go beyond hosts, towards networks and consistent services.

OSMT is free glue for free software. The glue has the policies and dimensions.

## 4.10 Solutions Based On Policies

Consistent solutions can be based on policies on top of capabilities.

## 4.11 The Complete Free Services Picture

The primary beneficiaries of the free software and free protocols concept are the service providers who deliver subscriber services based on free software and free protocols.

Users/Subscribers want solutions. Free software needs to be adequately framed and delivered to be of value.

Such delivery often involves matching/marrying free software with corresponding free services.

Here we describe the free model as we propose it.

There are four arenas of control, as shown in Figure 4.3. These are:

1. User Environment
2. Subscriber Server in User Environment
3. Subscriber Services Environment
4. User governed - Service Provider Premise Equipment

### 4.11.1 User Environment – Free Subscriber Environemnt

- Desk Top
- Lap Top
- PDA Environment (Linux PDA, WinCE, PalmOS, etc.)
  
- EOE on desktop
- 802.11 on Laptop
- WWAN on PDA – LEAP

### 4.11.2 Subscriber Environment Premise Equipment

Current examples are limited to layer 2 and layer 3 agents. ATTBI's modem. Cisco routers, etc.

### **4.11.3 Free Service Provider Environemnts**

Linux, Solaris, occasionally and tactically NT.

### **4.11.4 Service Provider Environment Subscriber Premise Equipment**

## **4.12 Free Services Software Perspective**

### **4.12.1 Subscriber Environment**

### **4.12.2 Service Provider Environment**

Bring over the software picture.

## **4.13 How tangibles Fit Together**

### **4.13.1 EOE's Place**

### **4.13.2 OSMT's Place**

### **4.13.3 ByName's Place**

### **4.13.4 LEAP's Place**

LEAP Manifesto [?]

### **4.13.5 MailMeAnywhere's Place**

#### **4.13.5.1 Architecture**

Figure 4.1 shows the architecture of Neda's Open Source Message Center on the server side. As shown in Figure 4.1, all of the available components can interface with the adopted software that is available outside the Neda software. For example, for OUTBOUND Mailers, Hyla FAX or Hyla PAGER software might be used.

#### **4.13.5.2 Server Side – Software for Service Providers**

Figure ?? shows the MailMeAnywhere Message Center Model.

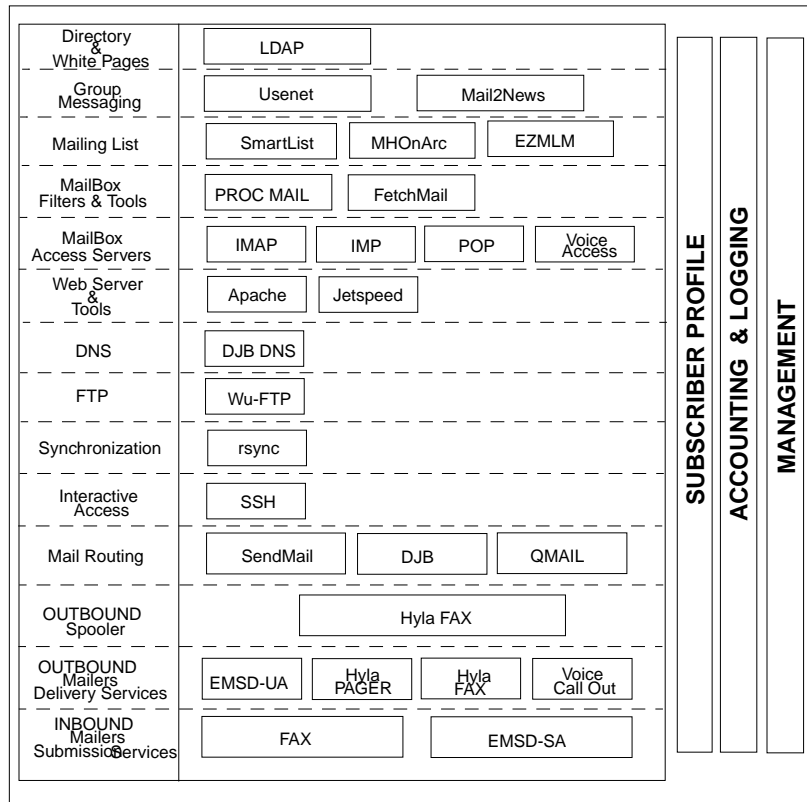


Figure 4.1: MailMeAnywhere – Server Side Software

#### 4.13.5.3 Linux

#### 4.13.5.4 Solaris

#### 4.13.5.5 Windows NT

#### 4.13.5.6 Client Side – Software for Users & Subscribers

Figure 4.2 shows the MailMeAnywhere Client Side Software.

#### 4.13.5.7 Linux

#### 4.13.5.8 Solaris

#### 4.13.5.9 Windows 95/98/NT

#### 4.13.5.10 Palm Pilot

#### 4.13.5.11 Windows CE

#### 4.13.5.12 Services

Figure ?? show the MailMeAnywhere Service Model.

### 4.13.6 The Role of Operation WhiteBerry

## 4.14 OSMT Overview

OSMT is the glue. Express policies. Enforce policies.

ksh

Why KSH?

Which KSH?

pdksh ksh88 ksh93 bash

Supported Environments.

linux solaris limited NT

Open platform libraries.

Software Type	Platform	Software Packages
Group Messaging	Win32	Netscape Communicator
	Sol2 / Linux	
Personal Filters & Forwarders	Win32	MailMeAnywhere
	Sol2 / Linux	
Mobile User Agents	WinCE	EMSD-UA
	Palm Pilot	EMSD-UA
	Win32	EMSD-UA
Desktop Mail User Agents	Exchange	OutLook
	Emacs	VM    BBDB    SuperCite    GNUS
	Win32	PINE    Netscape Communicator
	Sol2 / Linux	PINE    Netscape Communicator

Figure 4.2: MailMeAnywhere – Client Side Software

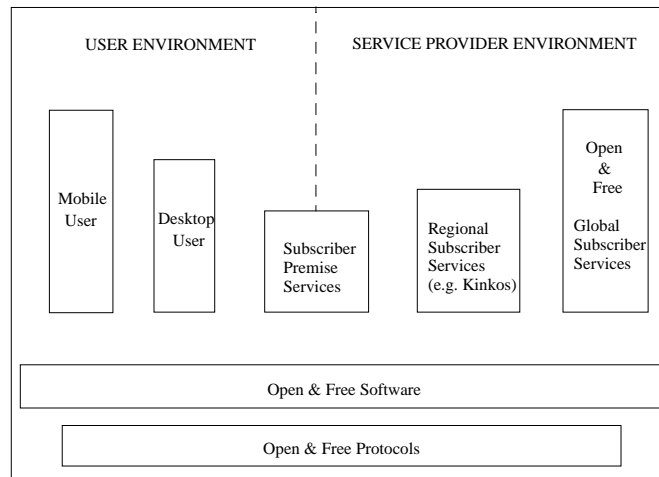


Figure 4.3: Open Services Management and Administration Platform

## **4.15 Central Modelling and Management**

### **4.16 Site and Cluster Switching**

### **4.17 Backup and Replication**

#### **4.17.1 OS**

#### **4.17.2 Rsync Update**

#### **4.17.3 Backup**

#### **4.17.4 CVS**

## **4.18 Configuration**

opConfig.sh, opBases.sh, opBasesSite.sh



## **Part II**

# **Subscriber Services: ByName and ByNumber**



## Chapter 5

# The ByName Service Components

### 5.1 Introduction

ByName and ByNumber services provide a lot of features for the subscriber. Upon the creation of these services for each subscriber (after receiving the confirmation back from the new subscriber), the new e-mail accounts and subscriber's home directory will be created instantly. All the features that he/she selected will be activated on the background, such as web pages and LEAP.

All of these task are run by the bynameNsp script.

Figure 5.1 shows the byname layer. Each of these layers will be explained in detail.

#### 5.1.1 Model and Terminology

NOTE: to make a more efficient way to create a new subscriber, the fullUpdate of bynameBatchBasic.sh is done in 2 different processes. The first process is the account creation (i.e. acctUpdate). This process have to be created immediately. The second process is the features activation (such as mailUpdate, webUpdtae, leapUpdate, etc.). The second process is run in the background.

Objects Overview:

```
-----  
item_bap_ai: bap_ Assigned Info.  
             The kind of info that we assigned such as  
             number, acctType, selector, initial passwd.  
  
item_bap_sid: bap Subscriber Identity.  
             The information about the subs' identity.  
             The subs can fill out this param freely.  
  
item_bap_sau: bap_ Subscriber Authentication  
             The subscriber is authenticated through  
             email, vouch, etc.  
  
item_bap_leap: bap_ LEAP feature information
```

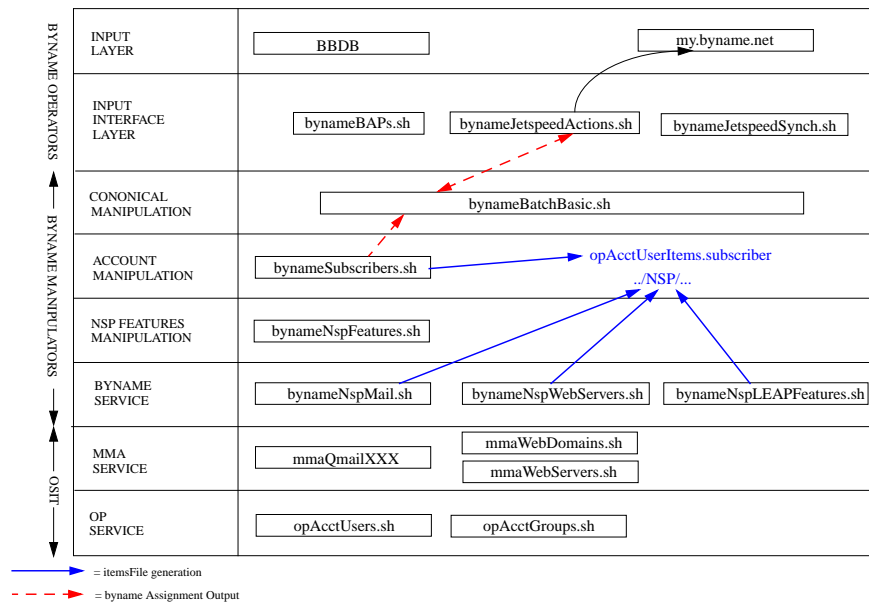


Figure 5.1: ByName Layer

item\_bap\_web: bap\_ Web Page info (e.g. style)

item\_bap\_mail: bap\_ mail feature info.

item\_sa\_SubscriberProfiles: the subscriber's profiles.

This information including login name and subscriber's id as well as personal info such as name, address, e-mail, phone, etc. This itemsFile is stored in each of subscriber's home dir in his/her NSP dir under the name bynameSubscriberProfiles.nsp.

item\_LEAPfeatures: config parameters for LEAP feature.

This parameters include his/her mobile device info, CDPD modem type and address.

item\_webFeatures: information about subscriber's home website.

This feature has not been used but will be in the future. It will contain information such as his/her web page flavor (e.g. professional, student, etc.).

item\_qmailAddr\_{acctName}: config parameters for creating the email acct. This support the forwarding information and which mbox the incoming message will be stored. General email acctName are:

- public
- personal
- office
- urgent

The non-standard email account can be any name requested by the subscriber. For example:

- records
- assistant
- fax
- admin

mmaQmail Object Processors and Containers:

-----

```

bynameBAPs.sh
bynameBatchBasic.sh
bynameJetspeedActions.sh
bynameNspSubscribers.sh
bynameNspFeatures.sh
bynameNspJetspeed.sh
bynameJetspeedHook.sh
bynameNspLEAPFeatures.sh
bynameNspMail.sh
bynameNspMailLists.sh
bynameNspTo97EMSD.sh
bynameNspWebServers.sh

```

## 5.1.2 At A Glance

Basic byname

-----

```

bynameBAPs.sh          -- BAP stands for Byname Account Parameters.
                        This script will read the itemsFile
                        for each of the subscriber.
                        The reason for having this itemsFile is to
                        be able to use the same information such as
                        id number, subsSelector, etc. and is most likely
                        for employee and their relatives.
                        The itemsFile is stored in a special dir
                        (e.g. ../nedaPlus/BAP/)
                        with the following filename convention:
                        acctTypePrefix.SubsSelector.Last.First.idNumber.BAP
                        (e.g. sa.1.simpson.lisa.20100.BAP)
                        The .BAP file contain all the parameters for
                        that particular subs. The parameters can contain
                        the following:
                        - User profile info
                          (name,address,phone,email,etc.)
                        - Features list
                          (leap,mail,web,etc.)
                        - Per feature parameters
                          * LEAP: deviceType, cdpdAddr, modemType, etc.
                          * Mail: cell phone number for text messaging,
                            main address, forwarding, etc.
                          * Web: style (stdent, prof, etc.)

                        After reading all of those info,
                        it will be passed to the bynameBatchBasic.sh
                        and will be processed as usual.

bynameBatchBasic.sh   -- the intermediate layer for the byname
                        creation. This also served as the
                        bridge from the web world to
                        the unix world.

bynameJetspeedActions.sh -- Called by Jetspeed to create the subscriber's
                        account from start to finish.
                        (e.g. setup uid, email, website, LEAP,
                        sending email confirmation, etc)
                        by calling other byname script.
                        This script is an interface between the
                        Jetspeed and byname tools.

bynameNspSubscribers.sh -- Create and update bynameSubscriberProfiles.nsp

```

```

saUID.nsp
bynameNspFeatures.sh      -- Manage all of the byname features in one place.
bynameNspJetspeed.sh     -- Has been absorbed into and rename
                           it to bynameJetspeedActions.sh
bynameJetspeedHook.sh    -- Called by Jetspeed to send out Emails and such
                           Has been merge to bynameJetspeedActions.sh
bynameNspLEAPFeatures.sh -- Creates LEAPfeatures.nsp file for the subscriber
bynameNspMail.sh         -- Creates/delete/update email account for subscriber.
bynameNspMailLists.sh    -- Track and control mailing list generation.
bynameNspTo97EMSD.sh     -- Generate EMSD Config files based on the
                           content of last/first
bynameNspWebServers.sh   -- Creates Virtual Web Server Domain entries
                           and initial webHome

```

## 5.2 Subscribers

### 5.2.1 bynameSubscribers

## 5.3 E-mail Accounts

### 5.3.1 bynameNspMail

### 5.3.2 bynameNspMailLists

## 5.4 Web Pages

### 5.4.1 bynameNspWebServers

## 5.5 LEAP

### 5.5.1 bynameNspLEAPFeatures

### 5.5.2 bynameNspTo97EMSD



## **Chapter 6**

# **The ByNumber Service Components**

### **6.1 Subscribers**

#### **6.1.1 bynumberNspSubscribers**



# Chapter 7

## ByName Web Service

### 7.1 ByName Web Architecture

Describe the relationship between IMP and JetSpeed and how to tie them with the OSMT and UNIX file system. Will be documented later.

#### 7.1.1 What is ByName?

#### 7.1.2 ByName E-mail Model

### 7.2 CVS Tree Checkout

The complete package of JetSpeed has been subject to CVS. Currently, you can check out by running this script:

```
/usr/devenv/mapFiles/byname/web/devel/current.sh
```

### 7.3 What is Jetspeed?

The my.byname.net is based on Jetspeed implementation.

Jetspeed is an Open Source implementation of an Enterprise Information Portal, using Java and XML. A portal makes network resources (applications, databases and so forth) available to end-users. The user can access the portal via a web browser, WAP-phone, pager or any other device. Jetspeed acts as the central hub where information from multiple sources are made available in an easy to use manner.

The data presented via Jetspeed is independent of content type, This means that content from for example XML,RSS or SMTP can be integrated with Jetspeed. The actual presentation of the data is handled via ates XSL and delivered to the user for example via the combination of Java Server Pages (JSPs) and HTML. Jetspeed provides support for templating and content publication frameworks such as Cocoon, WebMacro and Velocity. Note that outside of regular browser Jetspeed also supports WAP devices.

Jetspeed helps you build portal applications quickly. The goal is to make Jetspeed a tool for both portal developers as well as user interface designers. Currently the focus is on providing developers with a set of tools that facilitates

building the base for the portal. With Jetspeed you can quickly build an XML portal and also syndicate your own content.

## 7.4 Jetspeed Templates

Most of jetspeed templates that we are using are written using Velocity hence the file extension being “vm” which stands for Velocity Macro.

Velocity is a Java-based template engine. It permits web page designers to reference methods defined in Java code. Web designers can work in parallel with Java programmers to develop web sites according to the Model-View-Controller (MVC) model, meaning that web page designers can focus solely on creating a well-designed site, and programmers can focus solely on writing top-notch code. Velocity separates Java code from the web pages, making the web site more maintainable over the long run and providing a viable alternative to Java Server Pages (JSPs) or PHP.

The reference web page could be found at: <http://jakarta.apache.org/velocity>.

### 7.4.1 New Account Template

Location: `/jetspeed/WEB-INF/templates/vm/screens/html/NewAccount.vm`

This template create a form that a new user need to fill out to obtain a new user ID.

All of the fields that are used in this form have to be registered in the ExtInfo.java (ask YS to add the new variables) before you are able to used it.

For example the field FIRST\_NAME is already registerd in ExtInfo therefore we can use this as a reference in this syntax:

```
$ExtInfo.FIRST_NAME() -- as the input name
$User.get($ExtInfo.FIRST_NAME()) -- as the value of the input name
```

In HTML form context this will look like this:

```
<INPUT NAME="$ExtInfo.FIRST_NAME()" TYPE="TEXT" VALUE="$User.get($ExtInfo.FIRST_NAME())">
```

### 7.4.2 Anonymous ByName Front Page

Location: `/jetspeed/WEB-INF/psml/anon/html/default.psml`

### 7.4.3 Define Default Portal for New User

A new users home page is copied from the user TURBINE. Thus you should customized the TURBINE’s user home page to reflect the desired default home page for new users.

Location: `/jetspeed/WEB-INF/psml/user/turbine/html/default.psml`

### 7.4.4 Static HTML Place Holder

Location: `/jetspeed/byname`

### 7.4.5 E-Mail Confirmation and Verification

Location: /jetspeed/WEB-INF/templates/vm/byname/html/verification\_email.vm

Location: /jetspeed/WEB-INF/templates/vm/byname/html/confirmation\_email.vm

### 7.4.6 Template for Each portlet in the Anonymous Front Page

Location: /jetspeed/WEB-INF/templates/vm/portlets/html/byname-createNewUser.vm

Location: /jetspeed/WEB-INF/templates/vm/portlets/html/byname-login.vm

Location: /jetspeed/WEB-INF/templates/vm/portlets/html/byname-msgCenter.vm

### 7.4.7 Configure the layout and navigation bars

The look of Jetspeed's portal is controlled by the layout and navigation templates. Below are some of the file that define that look.

A Layout Manager is used in the generation of the resulting portal. Jetspeed support 2 Layout Manager, JSP and Velocity. Both layout manager produce the similar results, which one you use is dictated by which language you prefer.

Since we are using Velocity, the files uses by the Velocity Layout Manager are:

```
/jetspeed/WEB-INF/template/vm/layouts/html/default.vm
-- Display the logo in the upper left hand corner and
   define the top, bottom, and left navigation bars.
```

```
/jetspeed/WEB-INF/template/vm/navigations/html/top.vm
-- Top navigation bar when the user it NOT logged in.
```

```
/jetspeed/WEB-INF/template/vm/navigations/html/left.vm
-- Left navigation bar for all users.
```

```
/jetspeed/WEB-INF/template/vm/navigations/html/bottom.vm
-- Bottom navigation bar for all users.
```

To customize/update the navigation bars, follow these steps:

- Replace the reference to the old logo with new logo in default.vm.
- Add "banner adds" to top.vm.
- If no left navigation bar is desired, then remove the referance from default.vm.
- If a left navigation bar is desired, then update left.vm.
- Add any copyright notices, disclaimers, logos, ... to bottom.vm

## 7.4.8 Portlet Registration

Any site specific portlets should be define in the local-portlets.xrg. Everytime a new portlet is created it has to be registered in this file.

Location: /jetspeed/WEB-INF/conf/local-portlets.xrg

### 7.4.8.1 Examples of Registry Entry

Below is the example of page that is created using Velocity template.

```
<portlet-entry name="MsgCenter" hidden="false" type="instance" application="false">
  <meta-info>
    <title>Message Center</title>
  </meta-info>
  <classname>org.apache.jetspeed.portal.portlets.VelocityPortlet</classname>
  <parameter name="action" value="portlets.MsgCenterAction" hidden="false"/>
  <parameter name="template" value="byname-msgCenter" hidden="false"/>
  <media-type ref="html"/>
</portlet-entry>
```

Below is the exampel of portlet that are created directly from its static html file.

```
<portlet-entry name="byname-frontpage" hidden="false" type="ref"
  parent="HTML" application="false">
  <url>/byname/byname-frontpage.html</url>
</portlet-entry>
```

## 7.5 ByName Functionality Specification

### 7.5.1 Ideas and Bugs Report

Used this section to jot down any ideas that are need to be implemented for the next release or reported any known bugs that are need to be fixed.

#### 7.5.1.1 Ideas – Future Implementation

- Add ByName Announcements to the default page for each user’s byname web page. For instance: Latest additions: (with a drop down menu) “LEAP WINCE 2.1 Available” etc.
- Add the Follow-Me Messaging Preferences: Wireless E-Mail Forwarding Rules. This will be the portlet title. Set Mode: (drop-down menu) Mobile, Office, Home. On the same line add Wireless E-Mail Notification: (drop-down menu) LEAP, SMS, All, None.

The idea is that if someone set the mode from Mobile to Home, the Message Center’s MBOXes address will change accordingly. For example if the mode is Mobile, MBOX1 will only contain mobile@ and pager@ addresses, etc.

- At this moment, the e-mail confirmation and verification are sent out using the JetSpeed mail sent out system (perhaps with the javascript). Later we want to control this e-mail sentout by running the byNameNsp-MailSentOut.sh from the OSMT with the -p parameters.
- Additional services for existing subscriber will be created through an interview process. Take for example the TurboTax where you go through an interview process to gather the info and at the end the form format will be showed to you with all the info entered.
- Have the top page different for each page. For example instead, the anon page will have “Welcome to My.ByName” and the create new account page will have title “Request a New ByName Account”

### 7.5.1.2 Known Bugs

- If somebody fill out the form for creating a new account and not complete (i.e. the e-mail is not supplied) the error will be displayed correctly as expected. But if the second time another required field is not supplied, the error is not displayed instead it will go to the confirmation page. In other word, if you keep click on the Create New Account button twice, it will always go to the confirmatin page.
- In the Login screen, if you just click on Login button and nothing entered, the error page did not display anything. Or if any of the field was entered incorrectly, no errors warning...

## 7.5.2 State Table

The logic of creating a new account can be described in the state table. The description of the state table (figure will follow later):

1. State 1: Anonymous My.ByName front page. User Click: Sign up now, it will go State 2.
2. State 2: Applying mode. This will be the create new account page. When all the fields are entered and the Submit button clicked, go to State 3.
3. State 3: Verifying Data Input. If all field are entered correctly, confirmation Email1 will be sent out and continue to State 4. If one or more fields are not entered correctly, go back to State 2 and it will continue this way untill all the fields are verified OK.
4. State 4: Waiting for Confirmation. If the user does not respond after a certain period of time or if the user click on the RejectURL page (from Email1), continue to State 5. If the user click on the ConfirmURL page (from Email1) or he/she entered the secret code in the confirmation page, continue to State 6.
5. State 5: Account Denied. When account is denied, no actions are taken. Report the log and notices.
6. State 6: Account Created. When new account is created, Email2 is sent out. The user will receive his/her id number and the domain name.

The logic of login for existing account can also be described in the state table. The description of the state table (figure will follow later):

1. State 1: Subscriber Login Page. There are 2 methods of login: id and password or domain and Password. Click on the Login button and it will continue to State 2.
2. State 2: Validation. If id and password method is failed, go to State 3. If domain and password method failed, go to State 4. If validation success, go to State 5.

3. State 3: id and password Failure notice and then go back to State 2.
4. State 4: domain and password Failure notice and then go back to State 2.
5. State 5: Subscriber's web Page is displayed.

## 7.6 External Hooks

### 7.6.1 OSMT - byNameNspJetspeed

Within this section, we'll explain in detail the process that are performed by the `bynameJetspeedActions.sh`. The `bynameJetspeedActions.sh` calling other external script to do the job.

There would be 2 kinds of actions: create a new a account and remove account.

#### 7.6.1.1 Create a New Account

There are three ways to create a new byname account:

1. From the web (`my,byname.net`)
2. Batch entries (with `bynameBatchBasics.sh`)
3. Byname Account Parameters (BAP) file

Creating new account for byname is traditionally done through the web. When the new subscriber fill out all the information and click the "Create New Account" button, the `bynameJetspeedActions.sh` with the `-i enroll` is executed.

```
ACCOUNT PROCESSING -- Create New Account
```

```
=====
```

```
The outcome:
```

```
-----
```

```
Step 1:
  - assign new subscriber's selector
  - assign new subscriber's ID
  - create subscriber's home dir
Step 2:
  - generate subscriber's profile in
    <subsHomeDir>/NSP/bynameSubscriberProfiles.nsp
  - create a bynumber symlink
```

```
External hooks:
```

```
-----
```

```
Both steps are performed by calling bynameNspSubscribers.sh
with -i addNewSubs.
```

```
- assign new subscriber's selector
First check to see if the subs is already in by calling
bynameLib.sh's function: SubsIsAlreadyIn.
If subs is exist then exit otherwise the selector is assigned.
```

- assign new subscriber's ID  
The ``opUidAssignments.sh -s uid\_subscriber -a checkForNextUserID`` is called to get the next new subs id.
- create subscriber's home dir  
Now that we got bot SubsSelector and new SubsId, we are ready to create a new subs acct by calling opAcctUsers.sh.
- generate subscriber's profile  
There will be 2 new files in <subsHomeDir>/NSP dir:  
bynameSubscriberProfiles.nsp and saUID.nsp.
- create a bynumber symlink

## FEATURE SAVE

=====

After done with ACCOUNT PROCESSING, all the extra byname features will be generated. At this time, there is only one feature available: LEAP features.

The creation of LEAP feature was done by calling bynameNspLEAPFeatures.sh with -a generateNewNetL3ItemsFile.

## Mail out 1

=====

The mail out 1 is the mail verification sent out to the future subscriber. They need to reply to this email in order for us to activate the account. This mail verification is sent out only if it's done through a normal procedure (through the web). In this case the parameter genMode is empty.

If the genMode is specified and the value is ``batch`` then mail verification is not necessary.

## FEATURE ACTIVATE

=====

After the new subs verify the account, all the features are activated. This include creating the subs public web domain and all the mail creation.

The public web domain is created by bynameNspWebServers.sh with -a createSubsWebDomain. This process is run with remote shell through ssh to the public web server.

The mail creation is created with bynameNspMail.sh with -i acctAdd

## Mail Out 2

=====

Mail Confirmation is sent out to new subscriber which include all the information of his/her new byname account.

### **7.6.1.2 Delete an Existing Account**

Deleting an account is done through steps: delete subscriber's public web domain, delete all byname emails, and deleting his/her home dir and the username from `/etc/passwd`.

## **7.7 Plugs for IMP**

## **Part III**

# **User Environment**



## **Chapter 8**

# **Introduction**

NOTYET



## Chapter 9

# Emacs Office Environment (EOE)

### 9.1 About EOE

This publication applies to E-O-E – Emacs Office Environment as implemented for systems equipped with GNU-Emacs.

E-O-E is a collection of generalized commands, programming tools, software libraries, and related publications. Typically, an application programmer, a system manager, and a casual end user will require access to different subsets of this collection.

This manual is expected to function as a *Roadmap*. It enumerates the collection and the integration facilities that make the collection a cohesive environment. Each element of E-O-E is described in some detail. Where appropriate, reference to other sections of this publication or other related publications will be made.

Anyone coming to E-O-E for the first time will find it useful to read through an overview of concepts and facilities.

#### 9.1.1 About The E-O-E Package

The E-O-E Package is distributed through <http://www.mailmeanewhere.org/>

#### 9.1.2 Basic Office Services

All office workers independent of their specific disciplines need a set of "Basic Office Services". Electronic Mail, Time Management, On-Line Dictionary, Thesaurus, Personal Phone Book and Corporate Phone Book are examples of such Basic Office Services.

These generic office services are independent of the specific nature of the discipline (Accounting, Programming, Engineering, Legal, ...) that office worker is associated with.

These basic services are often provided at the workgroup level. In large organizations, many solutions to address these basic office requirements co-exists.

#### 9.1.3 Emacs Office Environment

GNU Emacs is an advanced, self-documenting, customizable, extensible, real-time display editor. The underlying part of GNU Emacs is written in C and includes a Lisp interpreter. Most of editing commands in Emacs are written in Emacs Lisp (elisp). Elisp provides for practically unlimited extension of GNU Emacs.

GNU Emacs runs on 100s of hardware/software platforms. Emacs runs on almost all flavors of UNIX. VMS, TOPS-20, ... are among other operating systems that Emacs has been ported to. GNU Emacs provides a unified level of service adequate for providing most basic office services on many hardware/software platforms.

What is needed to augment standard distribution of GNU-Emacs to a complete office environment is a set of co-operating pieces of software that are available on various ftp sites on the internet. Difficulty of locating the right version. Porting and configuring it your target environment. The importance of matching set.

There are many strategic advantages in using emacs as your "Office Environment". Here is a partial list of some of the advantages.

1. Uniformity of access to "Basic Office Services" through a Consistent User interface across diverse hardware/software platforms.
2. Hardware/Software Vendor Independence.
3. Open and extensible.
4. Integrated and Consistent.
5. Portable.
6. X-Windows and Character Based.
7. Conservation of Skill Sets.

E-O-E is a very rich environment and is targeted to sophisticated users.

Solving the problem once inside emacs and then using other programs from within it.

## 9.2 Configuration

Order of setting the parameters is:

pkg: The pkg itself sets the variables.

ee: The ee.

byname: bp-bynome-

site: bp-bynome-

group: bp-bynome-

user: bp-bynome-

## 9.2.1 Site Customization

### 9.2.1.1 eoe-load

### 9.2.1.2 eoe-require

## 9.3 Overview

## 9.4 Overview

### 9.4.1 Categories of Services

Through out this manual we categorize the Basic Office Services into three categories.

#### 1. Golbal Services.

These are capabilities that you want to have at your disposal independent of what you are doing. For example, You want to be able to run the spelling checker or the file completion capabilities when you are using your manipulating e-mail, entering calendar items, doing your desk top publishing or writing code. You want your spell checking and file completion capabilities to work uniformly independent of the specific task that you are doing.

- (a) Spell Checking (ispell)
- (b) Dictionary Look-up (webster)
- (c) Thesaurus
- (d) File Name Completion (filec)
- (e) Printing (lpr-buffer)
- (f) Corporate Phone Book (finger)
- (g) Personal Phone Book (rolo)
- (h) On Line Help (manual-entry, info)
- (i) Desk Top Management (window-)

#### 2. Generic Services.

These are the individual capabilities expected in the general office. In Emacs, these are typically associated with major modes.

- (a) Interpersonal Massaging (Rmail, MH, GNUS)
- (b) Time Management (Calendar)
- (c) Information Retrieval (info)
- (d) Unix Shell (cmushell)
- (e) Calculator (calc)

#### 3. Specific Services.

- (a) Software Development (C-mode, C++-Mode, Lisp-Mode, GDB)
- (b) Desktop Publishing (Tex, LaTeX, TeXInfo, LaTeXInfo)



## **Chapter 10**

# **Web-Based User Environment**

NOTYET



## **Chapter 11**

# **Linux User Environment**

NOTYET



## **Chapter 12**

# **Closed But Common User Environment**

NOTYET



## **Part IV**

# **Software Distribution Center: MailMeAnywhere**



## **Chapter 13**

# **MailMeAnywhere Open Source Software and Documentation Distribution Center**



## **Part V**

# **Commercial Support: Neda**



# Appendix A

## Neda Clusters

### A.1 Uncluster

### A.2 Office

System	Mail	DNS	Web	Exported Files
shoosh		192.168.0.10		
jamshid		192.168.0.103		i1 g1
yankee		192.168.0.105		j1
shiraz		192.168.0.106		
aladdin		192.168.0.107		
haji		192.168.0.108		
farhad		192.168.0.110		
gorz		192.168.0.123		
afrasiab		192.168.0.144		
sudابه		192.168.0.220		
zahak		192.168.0.251		

### A.3 Island

### A.4 DMZ

System	Mail	DNS	Web	Exported Files
roostam		198.62.92.1		
tooran		198.62.92.29		
arash		198.62.92.10		
majnoon		198.62.92.14		k1
tahmineh		198.62.92.32		

**A.5 Payk****A.6 Subscriber**

System	Mail	DNS	Web	Exported Files
yazd		198.62.92.46		

**A.7 Test****A.8 Public**