



The newest addition to the IBM Rational Developer for System z family -- The IBM Rational Developer for System z Unit Test feature, is based on the IBM System z Personal Development Tool (zPDT). RDz Unit Test is an add-on to the RDz integrated development environment, creating a personal System z development and unit test environment for a developer running a Personal Computer.

The new release of RDz Unit Test feature revolutionizes the way organizations develop and test mainframe applications, enabling a virtual System z architecture environment for development and unit test that allows mainframe operating systems, middleware, and software to run on Intel and Intel-compatible platforms. No System z mainframe hardware is needed for the *initial* set of development activities.

The IBM System z Personal Development Tool (zPDT) provides one or more System z processors (with several emulated I/O device types), based on a personal computer Linux environment. As the name implies, it is intended for development and related purposes, such as education and demonstrations. It lacks the RAS or quality of service functionality innately built in the System z machine and is not intended for production use. The IBM machine type identified with the System z Personal Development Tool is 1090.

IBM has encouraged the use of several very small S/390® environments for use by the IBM development community (IBM PartnerWorld® for Developers), and these have proven extremely useful. The 1090 offering provides a number of functions that extend the usefulness of very small System z development machines; these include the following:

- * More than two gigabytes of System z memory
- * Full 64-bit System z operation
- * QDIO channel operation
- * OSA-Express2 functions
- * The more recent instructions that have been added to System z processors
- * SCSI-attached tape drives, plus conversion utilities
- * zAAP, zIIP, and IFL processors
- * Simple installation and operation
- * Cryptographic adapter functions
- * Channel-to-channel (CTC) operations

Providing these functions does not produce an environment equal to a larger System z, of course. Some aspects of a larger system are unlikely to be met in any very small environment; these include the ability to verify and enhance the scalability of a program under development, run application programs that require hundreds of MIPS, or exploit cross-LPAR functions. A larger System z is needed for these areas of development. Likewise, a 1090 system is not recommended for very fine-level performance tuning that is sensitive to memory location, cache functions, and pipeline optimization; larger System z machines have different characteristics than a 1090 at this level. For these and other reasons, the 1090 is *not* intended as a *production* system.

The basic 1090 offering consists of the 1090 software (processor functions, device emulators, utilities) and a hardware key device that is installed in a USB port of the host processor. The hardware key determines the 1090 model that is used. The hardware key must be present (in the USB port) when the 1090 is being used, but may be removed at other times.

Three 1090 models are available: L01, L02, and L03. The model number indicates the number of System z CPs that may be defined and used by the 1090. In most cases, the underlying Linux PC (that is used to install and run the 1090 system) must have at least as many PC processors as the 1090 model number.

Several 1090 versions are available for different environments:

- * A 32-bit version for selected releases of openSUSE
- * A 64-bit version for selected releases of openSUSE
- * A 32-bit version for Red Hat® Enterprise Linux 5.3 or later
- * A 64-bit version for Red Hat Enterprise Linux 5.3 or later

The different versions for openSUSE and Red Hat are due to slightly different libraries on the two distribution bases. The 64-bit versions of the 1090 must run on the corresponding 64-bit Linux version, of course. The 32-bit versions of the 1090 can run on either 32-bit or 64-bit Linux distributions. The 32-bit and 64-bit versions refer to the Linux base operating system. All 1090 versions can run 64-bit System z code.

For distribution, the two 64-bit versions are packaged in the same distribution file. The zPDT installer program detects which version should be installed. Likewise, the two 32-bit versions are distributed in the same file.

The base Linux machine used for the 1090 must have sufficient memory. There is no specific size required, but 3 GB should be regarded as a minimum. Disk space is needed for emulated 3390 (or 3380 or FBA) volumes and a typical 1090 base machine will have at least 100 GB of disk space.

The 64-bit versions of the 1090 are recommended for new users. These versions offer better performance and larger System z memory options.

FUNCTIONS - 1090 functions include System z processor (CP) operation and the emulation of a variety of I/O devices. As a general statement, all the functions (instructions and I/O) needed to run current System z operating systems are provided. The CP instruction set is at Architectural Level Set 3 (ALS 3), plus additional instructions and facilities (generally, up through the z10 level).

System z character data is typically in EBCDIC, just as for any System z processor. Emulated disks and tapes typically contain EBCDIC data, although they logically contain whatever mix of EBCDIC, binary, ASCII, Unicode, or other formats that are produced by the System z operating system and applications. The key point is that there is no routine translation to the ASCII of the underlying host Linux system. The same binary data representation that is used on System z is also used on 1090 systems. This extends to fixed point, packed decimal, and all floating point formats. All 1090 data is in System z representation. There are exceptions for emulated card readers and printers, where the character set involved is relevant and conversions between ASCII and EBCDIC are needed.

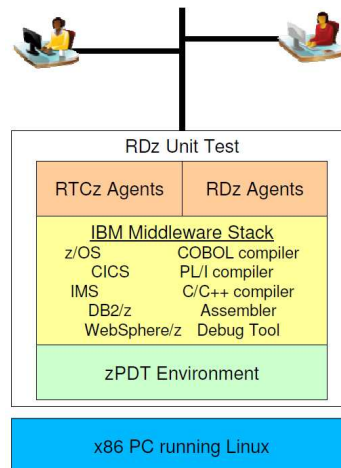
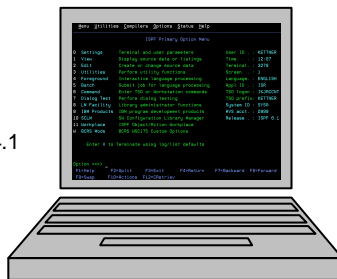
Not all System z instructions and functions are available with the z1090. Instructions related to specific hardware facilities or optionally used by specialized programs might not be present. This excluded list includes list-directed IPL, the accelerator function of cryptographic coprocessors, ETR, TOD steering, asynchronous data movers, MIDAWs, logical channel subsystems, Hipersockets, LPARs, multiple I/O paths, and Coupling Facilities. Not all CHSC functions are available. At the time of writing, the 256-bit AES functions were not present for the integrated cryptographic instructions; these may be added by subsequent maintenance releases for the 1090. The 1090 offering includes a number of *device managers*, each of which provides emulation for a related group of devices.

NOTE: The design of the 1090 allows for a large number of emulated I/O devices. The number is restricted, in practice, to better manage the memory and processing needed for emulated I/O. The 1090 currently allows a maximum of 1024 emulated I/O devices. This is often described as 1024 *subchannels*.

Principles - All current IBM System z operating systems (assuming proper licenses exist) are supported for 1090 usage. This includes current versions of z/OS, z/VM, and z/VSE. Linux distributions intended for System z usage may be used, but all functions and configurations have not been extensively tested. Older versions of operating systems and other software may work correctly (provided they are at least at the XA level), but there is no formal testing or support for older software.

Software installation methods may be different for 1090 systems than for traditional System z installations. This difference is due to the differences in I/O devices available on 1090 systems and on larger System z machines.

- Operating System**
- z/OS V1.10, V1.11
- z/OS Runtimes**
- IMS V10, V11
- DB2 for z/OS V9.1
- WS & MQ V7.0
- CICS TS for z/OS V3.2, V4.1
- Compilers**
- Enterprise COBOL V4.2
- Enterprise PL/I z/OS V3.8
- XL C/C++ V1.10, V1.11
- Assembler
- REXX
- Utilities**
- DFSORT
- SDSF
- IBM Debug Tool



Value Proposition - For System z customers who want to maintain, enhance, and create new applications for the mainframe. IBM Rational Developer for System z increases productivity of existing mainframe developers while appealing to new developers coming from a distributed ("open systems") background. It is the only product on the market that can offer a single solution to develop, test, and deploy an enterprise application from the display of HTML, to logic in WebSphere, to Web Services in CICS or IMS, to DB2 database access on the mainframe. We offer a development environment that integrates well into existing development practices using ISPF on the green screen while allowing for a transition to a twenty-first century development suite.

Target Audience - System z customers who want to improve the productivity of their core developers as well as leverage those existing skill sets in the company's efforts to create new applications for support of SOA.

This includes customers who:

- * Run traditional System z applications but wish to modernize and expose existing functionality in a service-oriented architecture
- * Talk about moving to composite architectures with both WebSphere Application Server and System z
- * Use traditional ISPF (green screen) development tooling but are looking for a higher-productivity development environment
- * Need tools to help their developers interface with application lifecycle tools or to enforce application governance
- * Want to attract new developers to the System z platform and quickly train them to develop applications
- * Want a single IDE to develop and debug an Enterprise application, from the Web interface, to the CICS and COBOL on the mainframe
- * Need to reduce the cost of application development for the System z platform, devoting more MIPS to production and fewer to development activities.

Some key usage scenarios include:

For any System z customers who want to lower development costs by reducing mainframe cycles used for development.

* By CICS developers, who:

- Are creating System z processing, including those that want to develop and integrate CICS and IMS processing in SOA
- Want to leverage the Web services capabilities provided in CICS V2, V3, V4, the Service Flow Runtime feature provided in CICS V3 and V4, or Service Component Architecture in CICS V4

* By IMS developers, who:

- Create System z processing, including those that want to develop and integrate IMS processing in SOA
- Create System z processing and services and leveraging the Web services capabilities in IMS and IMS SOAP Gateway

* By System z batch and UNIX System Services developers of COBOL, PL/I, C, C++, Java, and High-Level Assembler processing

* By WebSphere developers creating applications that deploy to multiple platforms including System z

* By EGL developers creating applications, Web UI, or Rich UI connecting to System z application logic.

Unlike other development solutions, IBM Rational Developer for System z can offer a single solution to develop, test, and deploy an enterprise application from the display of HTML, to logic in WebSphere, to Web Services in CICS or IMS, to DB2 database access on the mainframe. We offer a development environment that integrates well into existing development practices using ISPF on the green screen while allowing for a transition to a twenty-first century development suite.

The Unit Test environment provides a high fidelity, low-cost test environment for mainframe applications and runs actual versions of IBM middleware.

Contact IBM Rational Sales for further information.