From novice to expert in Dynamic Critical Path

(aka Workload Service Assurance)

written by Katiuscia Berretta, Senior Software Engineer and WSz Customer Support L3 Specialist

Table of Contents

Background

Using a Dynamic Critical Path (DCP) when badly configured can be really frustrating, as you might have experienced, and sooner or later will lead you to stop the feature and ask for help.

This paper is provided to introduce the terminology and to guide the configuration, so that you do not waste time and immediately identify all the actors having a role in the way the feature works.

That said, let us start this journey from the first steps.

To use Dynamic Critical Path (DCP) you have to consider the two components of the feature activity: *Modeling* and *Monitoring*.

Modeling:

- Defining Critical Jobs
- Time settings in applications and job definitions
- Configuration of DCP
- Configuration of related features.

Monitoring:

- Relying on EQQCPH* messages to see how well the critical workload execution is going
- Checking the hot list of each critical job to see if something in the critical network (i.e the network of ALL the predecessors for a critical job, at any level) is causing some delay to the execution of the critical job itself.
- Performing actions when needed or prompted (i.e. completing an operation, removing a dependency, restarting a job etc.)
- If the Dynamic Workload Console (DWC) is used, the dashboard gives a first glance of how the critical workload is proceeding.

Basic concepts, objects and terminology

Scope:

When an operation is set as a Critical Job (by specifying **P** as the Critical Indicator value in the Operation Automatic Options) the aim is to improve the monitoring and the possible actions to be sure that the job gets completed within its deadline. To do this, the monitoring involves the whole critical network (i.e. the network of all its predecessors at any level).

What:

The measure used to evaluate a possible risk for a critical job of not getting completed within its deadline is the **Estimated Start/End Time**. At daily plan batch run, it is initialized by the Planned Start/End Time. Then, during the plan execution, Estimated Start/End Times are updated dynamically according to what happens in the Current Plan (CP) and in the critical network.

How:

Among all the predecessors at the same level, the predecessor on the critical path is the one having the latest Estimated End Time. This means that more than any other predecessor at the same level it can put the execution of the critical job at risk. The path of these predecessors, level by level, is called the **critical path**.

The Hot List for a Critical Job

The **Hot List** for a critical job is the list of all the operations belonging to its critical network that are **Late** (i.e, still not started when the **Latest Start Time** comes), **Long Running** (i.e. in started status but not completed when the estimated duration has passed, according to long duration policies) or **Ended in Error**.

An operation exits the hot list when the condition for remaining there disappears: if it was late, it exits the Hot List when it gets started, if it was in long duration it exits the hot lists when it gets completed and if it was in error it exits the hot list when its status gets changed (from error to complete, ready).

Risk Levels for a Critical Job:

- **None** – Empty hot list and estimated end time not over the deadline
- **Potential** – The hot list contains at least an entry but the estimated end time is not beyond the deadline.
- **High** – Estimated end time is beyond the deadline. This is the original definition, later on in this document we will see how this definition can be completely customized.

Why is the critical path 'dynamic'?

The first critical path for each critical job is actually determined by the Daily Plan batch. In fact the Estimated Start/End times that decide if an entry belongs or not to the critical path of a critical job are initialized by the Planned Start/End times once a new CP is created.

However, during the current plan execution there might be changes and/or delays in certain points of a critical network that determine the need of recalculating the critical path for a critical job. In fact, during the CP execution, another path of critical job's predecessors might put at risk the critical job execution more than the initial critical path.

So the critical path is considered dynamic because it can change to reflect what happens to the critical networks during the workload execution.

The possible reasons behind a critical path recalculations are two:

- Modify Current Plan (MCP) actions affecting the critical network
- Accumulated delays in a branch of the critical network

What (on earth) does "accumulated delay in a branch" mean?

When an entry remains in the hot list for a quite long time, it accumulates delay in that branch of the network because it does not start (if it is late) or its successors cannot start (if it is in long duration or in error).

The critical network reflects changes in the CP but it is not updated for each change. In order not to affect performances of the controller, DCP works to always find a balance between accuracy (reflecting updates, delays etc) and performances (making only what is strictly needed).

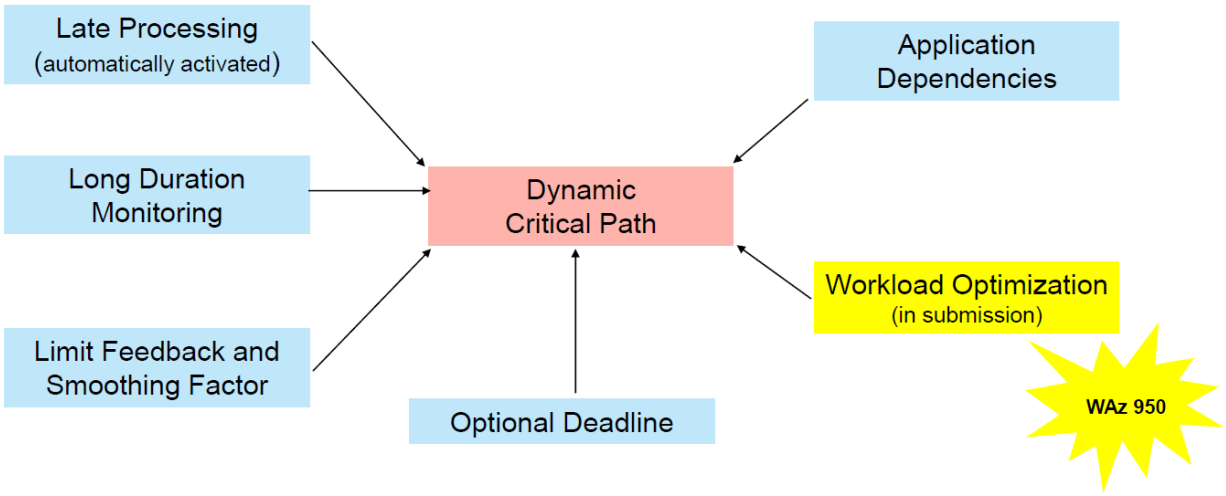
When an entry is in the hot list, nothing is done on the critical network until the accumulated delay really puts at risk the execution of the critical job more than the current critical path.

There is a timeframe of 'tolerance' for the accumulated delay (handled by internal structures, called TIEs, that act like clock alarms) until it becomes big enough to make that network branch more critical than the current critical path. When this happens, and only then, Critical Path Handler (CPH) task updates the Estimated Start/End times of that branch (not the entire critical network, always for performance reasons) by adding the accumulated delay to each entry of the branch and recalculates the critical path.

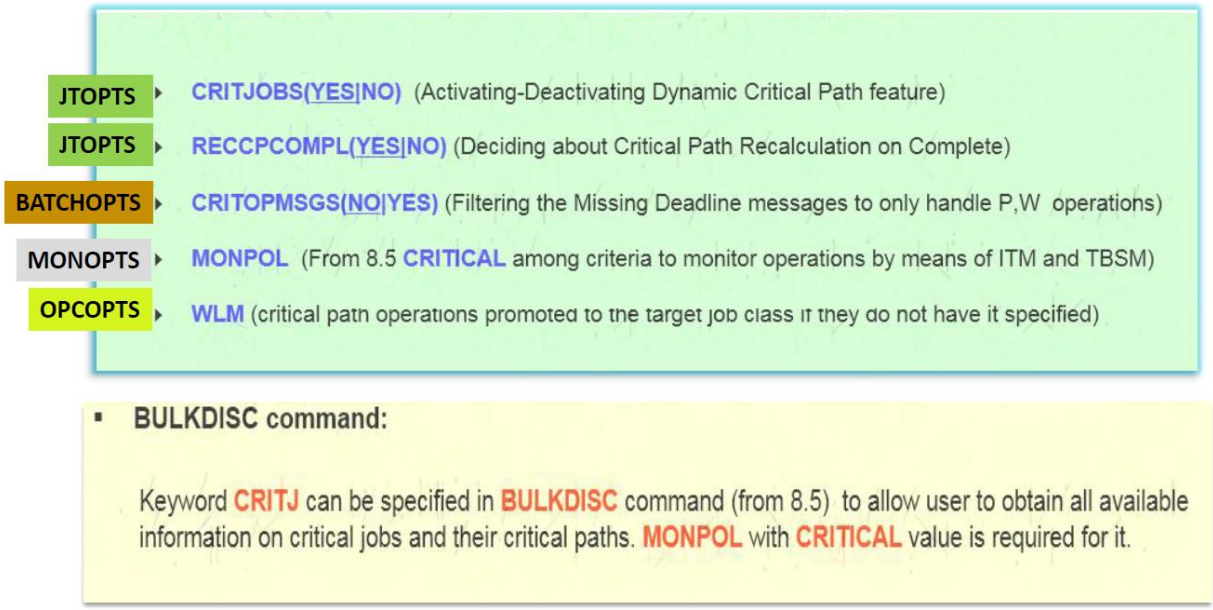
To summarize, the update of the Estimated Start/End times and the critical path recalculation are not triggered when an operation enters the hot list (late, long duration or error). It is triggered only when the accumulation of the delay is big enough to make another path more critical than the current one.

Dynamic Critical Path interacts with (almost) everything

Maps of the features supporting Dynamic Critical Path



Basic configurations: the easy part of DCP



By default, DCP is active (**CRITJOBS(YES|NO)**) but to have it really starting and working in your system, you need to have at least a critical job that was added to the CP via Daily Planning batch. In other words, this feature cannot be dynamically activated.

To activate DCP, once verified that there is at least one critical job added to the CP via DP batch, if CRITJOBS is set to YES the Normal Mode Manager (NMM) controller task creates the data spaces that will contain the entries corresponding to critical jobs and the whole networks of their predecessors. Then the NMM starts the Critical Path Handler (CPH) task of the controller that applies changes to critical networks reflecting changes to the CP.

DCP starts (data space creation, CPH get started) at controller start and it ends when DP batch produces a new CP (NCP). If the NCP contains at least a critical job, then once the new plan is taken over, the NMM task will start the DCP again (dataspace creation, CPH get started) based on the new data.

How to set a job as critical

You can define a job as critical but you can make a job become critical when it is already in the CP. You only need to specify **P** in the *Critical Indicator* field of the *automatic options* for the operation. A critical job is also eligible for WLM promotion when the needed conditions occur but you might want to make it only eligible to WLM promotion without setting it as critical: in this situation you simply set the critical indicator to **W**. If you specify N, the default, you are saying that it is either not critical or not eligible for WLM promotion.

When the critical path is recalculated too much

It is important to notice that, by default, the critical path for a critical job is recalculated also for *complete* reason (**RECCPCOMPL(YES|NO)**), that is each time an operation residing on the critical path gets completed. This might affect performances determining a lot of recalculations for the critical path that generally are not needed. The recommended setting of this parameter is **NO**, although the default is YES. Some exceptions can be evaluated if the amount of data in the critical networks is small.

The best friend (or the worst enemy) of DCP: Latest Start Time

Latest start time is the latest time an operation can start to get completed within its deadline. When the latest start time comes and the operation has not started yet, the operation becomes *late.* And if it is a predecessor at any level for a critical job, it enters its hot list and makes the Risk Level become **Potential** if it was None.

Because of its crucial role in DCP, Late Processing feature, that is optional in WSz (it can be activated by the ALERTS keyword) is automatically activated when DCP is active. No matter what you specified in the ALERTS keyword, the internal late processing (not including late messages) is always active to decide if an operation is late or not, based on its Latest Start Time.

The Latest Start Time cannot be manually set, it is calculated by WSz. However, a correct time setting guarantees a realistic and effective calculation of Latest Start Time.

Latest Start Time calculation is based on deadlines and estimated durations. A wrong setting of one or both of these two values will make Dynamic Critical Path completely run out of control and be wrongly handled by the other tasks.

There are at least two valid reasons to have correct Latest Start Times:

- Reducing ***false alarms about late conditions (and Potential Risk Levels)***
- Increasing ***accuracy*** of monitoring and actions

False alarms in late conditions drive DCP crazy

A false alarm in the DCP occurs when an operation *A* belonging to the critical network of a critical job gets late (and consequently causes the critical job's risk level to become potential) while the deadline of the critical job does not justify this condition. Getting late means that the operation is not yet started when its **Latest Start Time** comes.

A false alarm about late conditions is always caused by one (or both) of these factors:

- Inaccurate definition of **Deadlines**
- Inaccurate definition of **Estimated Durations**

False alarm in late conditions: S.O.S. Deadlines

The most frequent reason behind a false alarm is that an **operation A belonging to the critical network of a critical job has a deadline that precedes the deadline of the critical job**. When this happens, operation *A* gets late according to its own deadline, rather than getting late based on the critical job's deadline, that is what we are monitoring.

For example, if the critical job has a deadline set to next week and operation A has a deadline today, perhaps expiring in some minutes, operation A very likely will become late. Consequently it will enter the hot list of the critical job and put the critical job in potential risk. And this would be a false alarm because the late condition for A is determined based on its own deadline rather than on the deadline of the critical job, that is next week.

Therefore, a predecessor for a critical jobs at any level must not have deadlines preceding the deadline of the critical job.

So there are only two possible ways to set the deadline of operation A:

- If we are also interested in operation *A*'s deadline, *A* must be set as a critical job as well.
- If we are only interested to critical job's deadline, then operation *A*'s deadline must be equal or later than the critical job's. This way any late condition will be calculated based on the deadline of the critical job, that is the one we expect DCP to monitor.

In other words, the real definition of a **critical job** is a job whose deadline is important and cannot be moved ahead for any reason. If the job is not critical, the deadline can (and should) be moved beyond the deadline of the critical job.

If you don't want to manually move all the deadlines beyond the deadline of the critical jobs you can automate the process. The **BATCHOPT** parameter **IGNOREDEADL(YES|NO)** allows you to have the deadlines for all the operations in the CP (no matter if added via DP batch or dynamically via MCP) moved to the **CP Tail End** (i.e. beyond the CP End, in order not to be meaningful for the plan execution) except for critical jobs and suppress-if-late operations. It is strongly recommended that this parameter is set to YES when using DCP, otherwise false alarms will prevent the feature from effectively monitoring the critical networks, you will observe hot lists full of entries that should not be there and potential risk levels that are not realistic. As an alternative to the **IGNOREDEADL** parameter you can rely on the **optional deadline** (you are allowed to leave the deadline blank when defining applications and operations) and be sure that only the deadlines for critical jobs are set.

When the "bad" deadline resides inside the occurrence

We recommend you keep in particular consideration the way you set deadlines inside an application. If a critical job has a deadline that is different from the deadline of the application it belongs to, be sure that it is NOT later than the application deadline, in order to avoid that the calculated latest start time is anticipated and generates false alarms.

False alarms in late conditions: when deadlines are not guilty

It is important to notice that even if deadlines are all correctly set, false alarms in late conditions might still occur. Latest start time depends both on deadline and on estimated duration, and it is influenced by successors. The reason behind a too anticipated Latest Start Time (that might determinate a false late condition) in a critical network where all the deadlines are correctly set, is that there is at least one operation among the successors having an **estimated duration** that is particularly long and not realistic. Since the Latest start time is calculated as

(Deadline – Estimated Duration)

and it is mediated with the Latest Start Times of successors, if an Estimated Duration is particularly long and not realistic, the resulting Latest Start Time for the operation is anticipated.

For example, if an operation B has a deadline tomorrow at this time and an estimated duration of 24 hours, in absence of successors its Latest start time is *now*. And any predecessor for B will have an earlier (or equal) Latest Start Time, even if their estimated durations are short.

By principle, the Latest Start time for an operation cannot come before the Latest Start Time of a predecessor, it means that if the Latest Start Time for an operation is anticipated, this will reflect on all the predecessors.

Coming back to DCP, even if the operation having a long and unrealistic estimated duration is a successor at any level of the critical job (and consequently not included in its critical network) it affects the latest start times of the whole branch of its predecessors

If something like this happens, the only way to come to a solution is to identify the operation that has a very long (and unrealistic) estimated duration. To do this you need to manually navigate via ISPF among the operations in the CP: please notice that I said CP, not data space, because the operation can be a successor for the critical job and be out of the dataspace. Starting from the operation you observed, the one that has the abnormal Latest Start Time and is determining the false alarm in the critical network, you should navigate among its successors, level by level, until you find an operation having an estimated duration particularly long and not properly realistic. Once, in a customer environment, we found a 99 hours estimated duration, set by mistake by an operator.

When searching for a successor having a long and unrealistic duration, the *all deps* command in MCP might help you have a look through the operations branch although the determination of the 'bad successor' is always manual, not automatic.

To prevent this kind of problems (long and unrealistic durations) we strongly recommend the use of **Limit Feedback** and **Smoothing Factor** on Durations to make the estimated durations as realistic as possible. You can also use the **FIRSTFDBK** parameter to have the actual duration replace the estimated duration the very first time a job runs.

You are also allowed to use Limit Feedback and Smoothing factor for Deadlines but we strongly recommend you do not for not critical jobs: as explained above, deadlines for not critical jobs should be absent or moved beyond the CP End.

In JTOPTS Limit for feedback and Smoothing Factor for durations (don't use it for deadlines):

LIMFDBK(100)

The LIMFDBK value determines if estimated durations in the application description are updated when an occurrence of the application reaches complete status.

Feedback values are in the range 100 through 999, or 0 if the duration must be always updated, regardless of the estimated and actual values. The feedback limits are calculated as follows:

Limits for duration feedback

```
Lower limit = OD * 100/LF
Upper limit = OD * LF/100
```

where:

OD The old estimated duration currently stored in the application description database.

LF The limit for duration feedback.

LF value	Result
100	No new estimated duration are stored in the application-description database.
110	The new estimated duration is stored if the actual duration is approximately between 90% and 110% of the old estimated duration.
200	The new estimated duration is stored if the actual duration is between half and double the old estimated duration.
500	The new estimated duration is stored if the actual duration is between one-fifth and five times the old estimated duration.
999	The new estimated duration is stored if the actual duration is between one-tenth and 10 times the old estimated duration.

SMOOTHING(50)

The smoothing factor determines how much the actual duration of an operation influences the new estimated duration that is stored in the application description database. The smoothing factor is applied only if the actual duration lies within the limits determined by feedback

Note: When the controller has the Dynamic Critical Path feature active, any SMOOTHING value that is greater than 100 is internally managed as if the smoothing factor default value was set (50).

```
New estimated duration
ND = OD + ((AD - OD) * SF/100)
```

where:

ND The new estimated duration to be stored in application description database.

OD The old estimated duration currently stored there.

AD The actual duration.

SF The smoothing factor.

Factor	Result
0	There is no feedback.
10	The new estimated duration is the old estimated duration, plus one-tenth the difference between the actual and old estimated duration.
50	The new estimated duration is the old estimated duration, plus one-half the difference between the actual and old estimated duration.
100	The actual duration replaces the old estimated duration.
999	The new estimated duration is the old estimated duration, plus 10 times the difference between the actual and old estimated duration.

FIRSTFDBK(YES,NO)

First feedback for duration. If you specify YES, every new job that you define in the AD database is updated with the actual duration at its first run, regardless of the estimated values. At the next run, the duration is updated according to the values that you set for LIMFDBK and SMOOTHING.

If accurate Limit Feedback and Smoothing Factor still produce inaccurate data

Another useful way to improve monitoring of critical networks is to set **variable durations, by run cycle**, if for example you know that during the week ends a job lasts more than during the working days (or viceversa). You can also set a job as critical only some days, at some times, because also the variable **P** indicator by run cycle is supported by WSz.

When DP batch is the "culprit": Estimated vs Planned Times

According to the original definition, a critical job is in HIGH RISK level if its **Estimated End Time** comes after the **Deadline**. When this happens, message EQQCP21 is issued in the controller MLOG.

For planned operations in the CP, **Estimated Start and End Times** are initialized by **Planned Start and End Times** (i.e. the times that DP batch calculates and set as a result of a simulation of the scheduling activity) respectively. Then, **Estimated Start and Ends** are dynamically updated by CPH task based on possible accumulated delays in the critical network (i.e. the network of all the predecessors, at any level, of the critical job) during the current plan execution.

When a critical job is in HIGH RISK, it is a good practice checking its Planned End Time (in the details of the operation in the CP). If it comes BEFORE the deadline, then the focus of the analysis should be CPH task. If instead it comes AFTER the deadline, then the focus of the analysis should be the Daily Plan Batch. In this second situation, you should check if there are incorrect time settings or configurations (e.g. unavailable special resources, closed workstations, pending predecessors and so on) affecting operations that belong to the critical network and make the needed corrections.

If for example a pending predecessor is part of a critical network (i.e. a not solved mandatory predecessor), DP batch will not able to assign a **Planned Start/End Time** to it because it does not know if and when the operation will arrive. So, these times are set to the **CP Tail End**. The same will happen to all its successors, at any level, including the critical job(s). This will determine that the critical job(s) will be always in HIGH RISK.

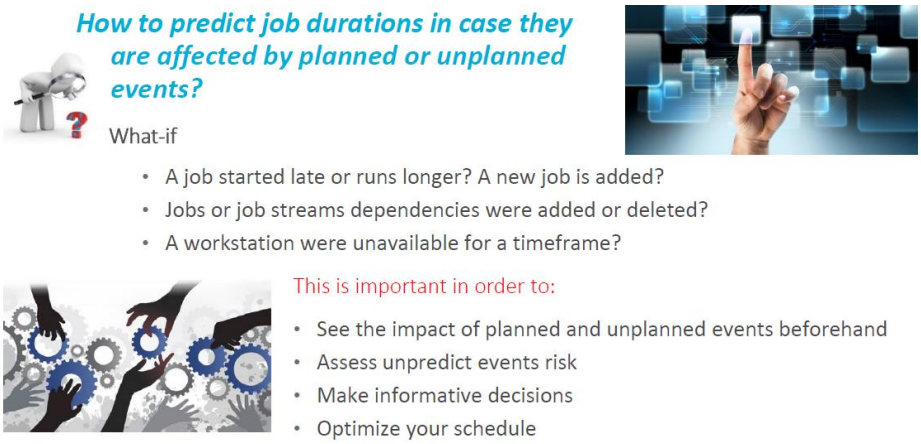
Time handling and *predict* capabilities for DCP

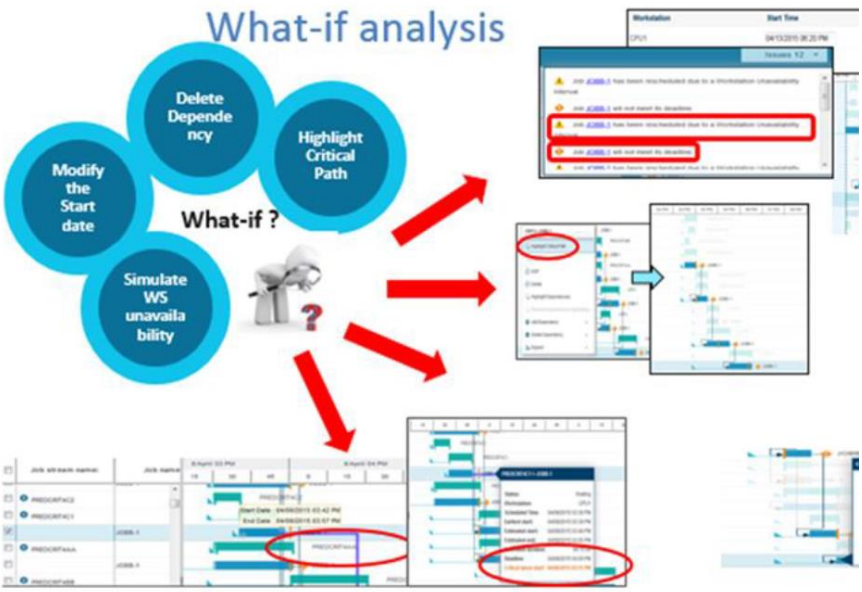
DCP predict capabilities mainly rely on three factors:

- ***What-If Analysis*** (manual predict) only available via DWC
- ***Variance*** for Estimated Durations and Estimated Start/End Times
- ***Confidence Factor*** for a critical job

What-If Analysis (DWC only)

You are allowed to have a preview of the consequences of possible changes in the CP, such as MCP actions on operations, on occurrences, on dependencies, on workstations, so that you can more easily see if your action can negatively or positively affect DCP functioning.





Variance exists but it is only used internally

For those who do not remember the term *variance, it* refers to a statistical measurement of the spread between numbers in a data set. More specifically, variance measures how far each number in the set is from the mean (average).

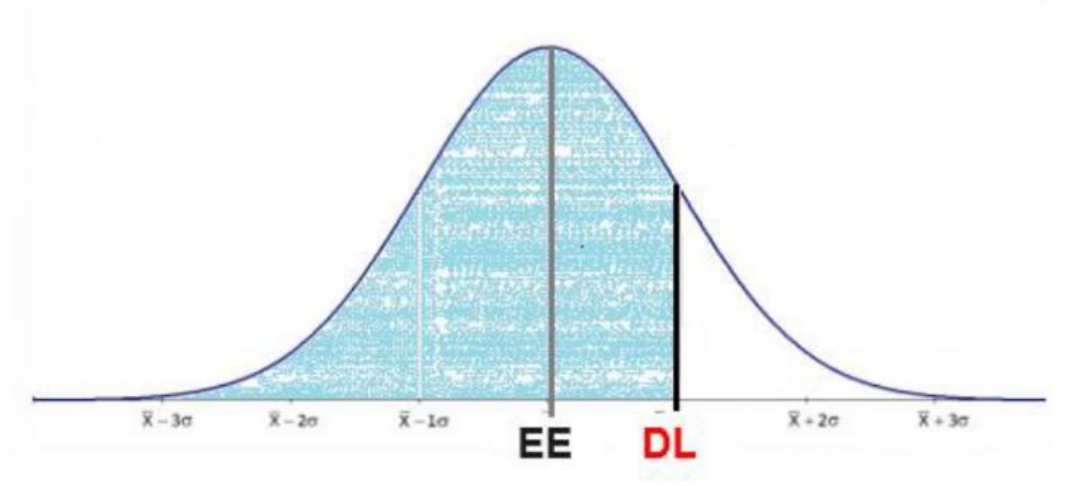
In WSz variance is calculated and set (not externally visible) every time the estimated duration for an operation is updated in the AD by Limit Feedback/Smoothing Factor. Based on that, variance is also calculated and stored by DP batch to determine the Planned Start Times for the CP operations.

When DCP starts after a DP batch run, Planned Start /End times initialize the Estimated Start/End times of the entries in the critical networks. The *Critical Path Handler* (CPH) task of the controller calculates the variances for the Estimated Start/End times each time they get updated to reflect changes in the CP or to process accumulated delays.

Variances cannot be either seen or used by operators or administrators. Their use is only internal and their focus is on determining the ***confidence factor*** for a critical job.

Confidence factor: that's why variance matters

The CPH task uses variances to calculate the **Gaussian distribution** of the **Estimated End Times.** This leads to the **Confidence Factor**, that is the probability for the critical job to get completed within its deadline (in the figure below EE=Estimated End Time, DL=Deadline).



The confidence factor is calculated by the CPH task for each critical job. If you are asking what you can do with the confidence factor, apart from checking it, the answer is here below.

Your own definition of high risk level for critical jobs

The original definition of high risk for a critical job is that the Estimated End Time is beyond the Deadline.

By using the **Confidence Factor** for a critical job, the definition of high risk level for a critical job can be reviewed and completely customized to become more precise and better fit your needs.

The **JTOPTS** parameter **RISKCONFIDENCE (1-99)** puts the Confidence Factor in a strict relationship with the High Risk Level for a critical job. It indicates a level of probability.

When the RISKCONFIDENCE parameter is not specified, the definition of High Risk is the original one (slightly corresponding to value 50).

If you specify RISKCONFIDENCE (70) you are saying that a critical jobs are in high risk when their confidence factor become smaller than 70%, no matter if the estimated end times go beyond the deadline or not.

The RISKCONFIDENCE option is only global, you cannot specify it at job level (i.e. you cannot specify different values of RISKCONFIDENCE for different critical jobs).

Please notice that if a critical job goes late (i.e. it has not started yet when its Latest Start Time comes) its confidence factor is automatically set to 0. But in a well configured system, this should never happen.

On critical paths, DCP does everything by itself

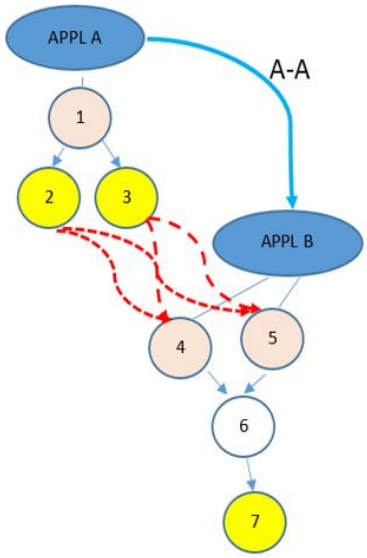
- ***Promoting operations that are NOT started and close to be late: the urgent queue***
  For jobs residing on the critical path that are starting late, a promotion on the urgent queue (maximum priority) is automatically performed in order to accelerate their submission. You don't need to configure CPH to do it.

- ***Promoting started operations: WLM promotion to a specified Service Class***
  Started operations residing on the critical path that are experiencing a long duration are promoted to a better performance Workload Manager Service Class if you have correctly configured the WLM interaction. You can globally specify a class (WLM Keyword) and a policy but you can also do it at operation level. A started operation on a critical path, compatible with the policy for promotion, is promoted to the class specified at its level, if any, or at the class specified for the critical job, if any, or at the global class.
  No automatic WLM promotion is performed on not started operations.

What happens to DCP if you use application dependencies?

Together with 'normal dependencies' that are dependencies between single operations (i.e. one operation is predecessor or successor for another operation) the scheduler also support *application dependencies*, when for example an operation depends on an entire application or if an application depends on an operation, or if an entire application depends on another entire application. To better clarify, there might be applications that are predecessors or successors for an operation or for another application. When a predecessor is an occurrence of an application it means that the subsequent workload can start only when all the operations inside the predecessor occurrence are completed.

Internally dependencies between two occurrences (e.g. A predecessor for B) are handled as dependencies between operations: all the First Operations (FOPs) of B are successors for all the Last Operations (LOPs) of A.



DCP supports application dependencies because in the critical network we can have application dependencies. However, according to the design illustrated above, all the calculations, the representation and the Estimated Start/End Times assignments are based on dependencies between operations.

What happens to DCP if you use conditional dependencies?

An operation A conditionally depends on one or more other operations (e.g. B,C,D) if a condition on B,C,D status or RC was defined. And when the condition becomes true. A can start.

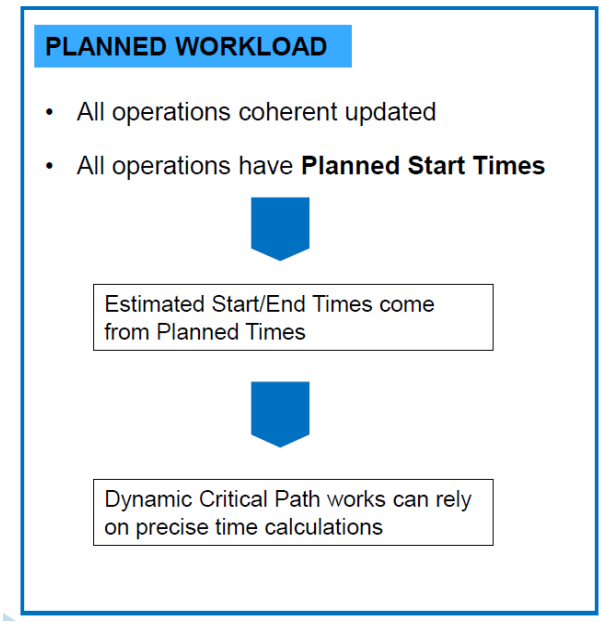
DCP does not support conditional dependencies, so possible conditional predecessors for a critical job at any level are not included in its critical network.

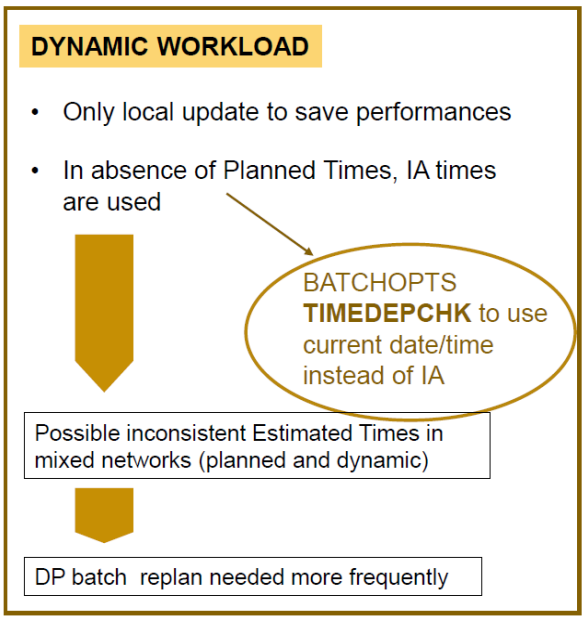
Planned vs dyn, amic-added workload in handling DCP

When running with DCP in your systems you need to know that there are significant differences between the workload that comes from the DP batch and the workload that is dynamically added (ETT, ISPF 5.1, PIF) to the CP.

The main reasons behind these differences are two:

- The Estimates Start/End Times are initialized by Planned Start/End Times and Planned Start/End Times do not exist for dynamic added workload.
- To save performances in MCP, some important updates occurring in the network after dynamic adds occur only inside the affected occurrence. For example, if the occurrence is included in a critical network, only the Estimated Start/End times inside the occurrence are set and updated based on internal dependencies. No update of the estimated times occur in the rest of the critical network.

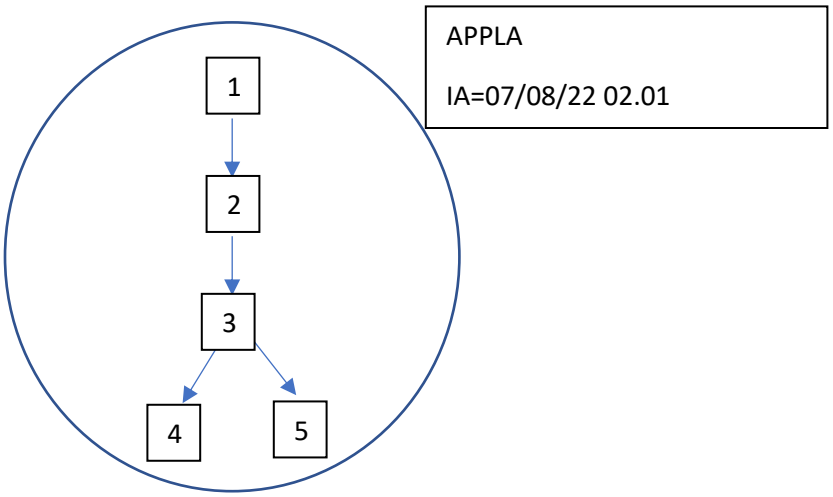




So, how can we use (without magic) our DCP in a WSz environment where both planned and dynamic-added workload are present?

First of all, what does CPH use to initialize the Estimated Start/End Times for dynamic-added workload since the Planned Start/End times do not exist?

By default, the occurrence **IA Time** is used to initialize the Estimated Start Times of the FOPs (first operations) of the added occurrence. The estimated End Times will be calculated by adding their Estimated Durations. The Estimated Start/End Times for the other operations in the occurrences are calculated by using the IA Time and keeping in consideration durations and internal dependencies. For example, if you are adding an occurrence of application APPLA, with IA=07/08/22 02.01, the estimated times of each single added operation will be the following:



Let us suppose that estimated durations are 1 minute for 1, 1 minute for 2, 5 minutes for 3, 1 minute for 4 and 2 minutes for 5.

The Estimated Start/End Times will be:

Operation 1 : EST: 07/08/22 02.01 EET: 07/08/22 02.02

Operation 2: EST: 07/08/22 02.02 EET: 07/08/22 02.03

Operation 3 EST: 07/08/22 02.03 EET: 07/08/22 02.08

Operation 4 EST: 07/08/22 02.08 EET: 07/08/22 02.09

Operation 5 EST: 07/08/22 02.08 EET: 07/08/22 02.10

However unless an operation is time dependent, the IA time, as a concept, does not appear related to the Estimated Start Time. The need of using something more precise came very soon.

If you don't want to use IA time, you can set the **BATCHOPTS** parameter **TIMEDEPCHK(YES|NO).**

If you specify YES and run a DP batch, then:

- the Planned start Times are calculated starting from the start date of the CP rather than from the IA Time unless operations are time dependent. This option was introduced because by default DP batch, when simulating the CP execution, handles all the operations as they were time dependent, and calculates the planned start times based on the IA. In fact, by default there is no operation having a Planned Start Time preceding the IA.
- for each occurrence added via MCP to the new created CP, the Current Date/Time will be used to initialize the Estimated Start Time of the FOPs (if they are not time dependent) instead of the IA. The processing to update the Estimated Times inside the occurrence is the same (keeping into consideration durations and internal dependencies).

As we can see, the use of the **BATCHOPTS** parameter **TIMEDEPCHK** makes the Estimated Start/End Times more precise than the default, based on IA, but does not resolve the main problem: **the update of the Estimated Start/End Times only occurs inside the dynamically added occurrence.** To save performances, no update is applied all around the occurrence even if it is part of a greater critical network. As a consequence, the massive use of dynamic adds might negatively affect consistency: we might find entries whose Estimated End Time comes after the Estimated Start Time of its successor, for example.

Always for performance reasons, if you manually change a deadline, the latest start times are recalculated by the MCP only inside the affected occurrence. And this can also contribute to affect consistency.

If you are used to a massive dynamic add of occurrences that are not "stand alone" (i.e. containing a critical job and forming an autonomous critical network rather than being embedded in a greater, already present, critical network), an increase of DP batch replans is recommended in order for the DCP to be restarted with coherent and consistent data. No matter the value you specified for TIMEDEPCHK, the reason behind the suggestion for this increase is that for dynamic adds the update of the Estimated Start/End Times occurs only for the operations belonging to that occurrence.

The DP batch replan will recalculate The Planned Start Times for all the operations in the CP and remove the completed occurrences. DCP will then start from coherent data.

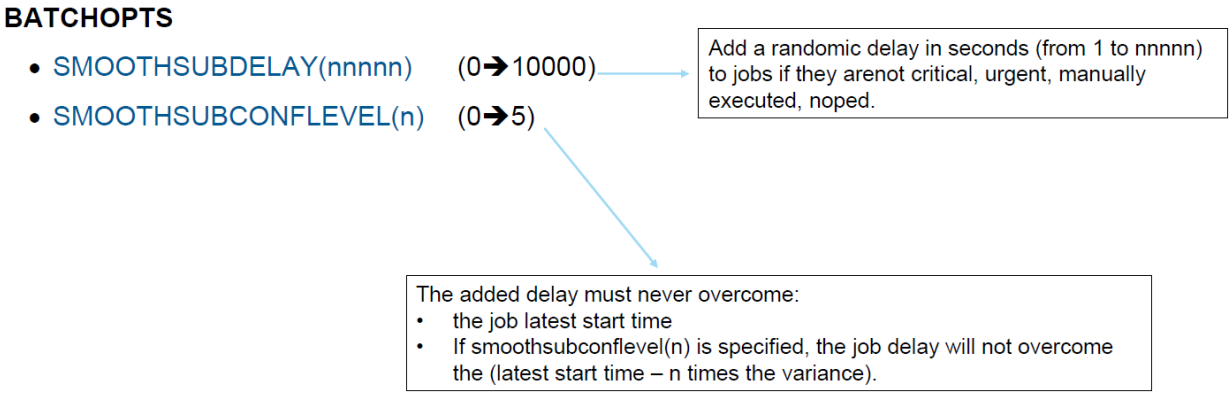
Smoothing the submission to enhance DCP capabilities

By default WSz submits as soon as possible all the jobs in the CP. For big workloads this might lead to system overload and job interlock.

The need to avoid the system overload and, at the same time, the need to prioritize the critical workload led to the development of the *Smoothing Submission* feature (also known as *Workload Optimization in Submission*). Basically we need a way to slow down the normal (not critical) workload and prioritize the critical workload.

Smoothing Submission is made of different configurations and different levels of complexity.

Adding a planned delay to all the jobs that are not critical

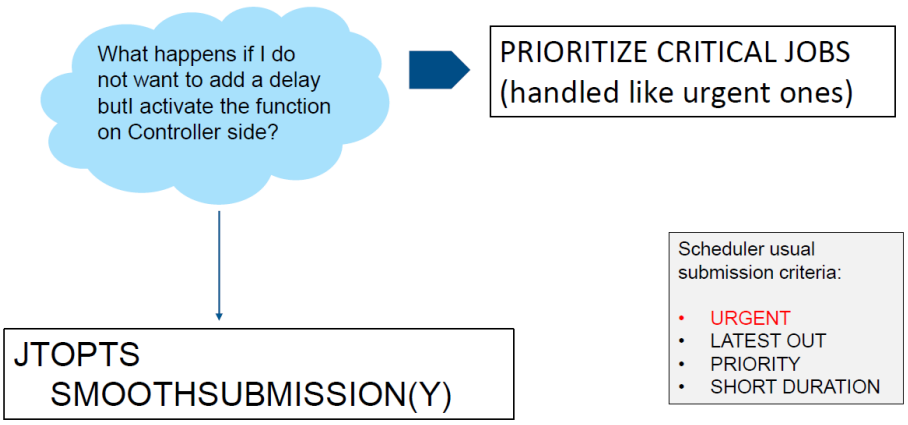


The above BATCHOPTS parameters don't apply any delay. They only make the DP batch update the CP01 (header record) so that the controller can know they are used. The *Smoothing Submission* must be activated from the controller side:

JTOPTS SMOOTHSUBMISSION(Y)

If you run the DP batch with the two parameters and the controller is running with SMOOTHSUBMISSION(Y) you will have the **critical jobs prioritized** and the **not critical jobs slowed down (by adding a delay)**.

If you don't set the BATCHOPTS parameters (SMOOTHSUBDELAY and SMOOTHSUBCONFELEVEL) but the JTOPTS SMOOTHSUBMISSION(Y) is set in your controller, then you will have only a **prioritization of critical jobs** (handled like urgent ones)

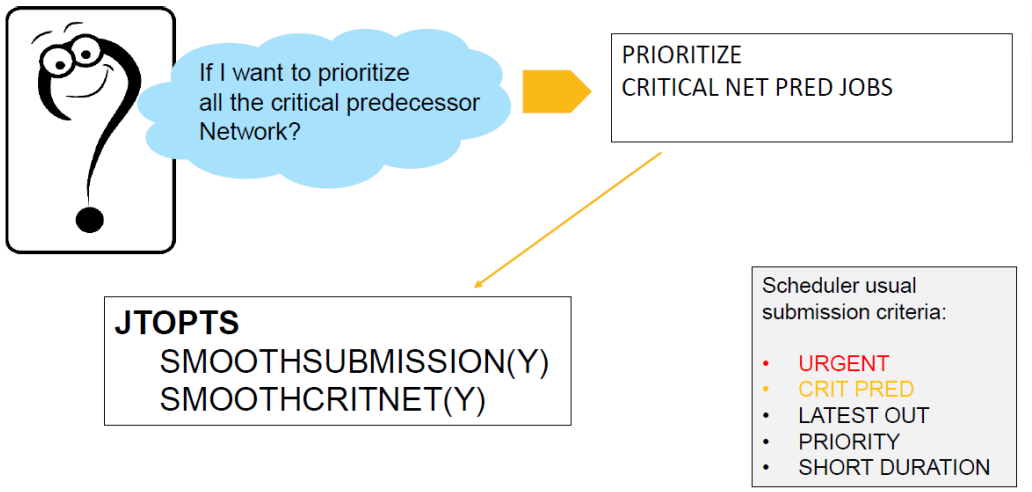


Great, you prioritize the critical jobs. And what about the critical network?

As we have seen above, to prioritize the critical jobs you have to specify

JTOPTS SMOOTHSUBMISSION(Y)

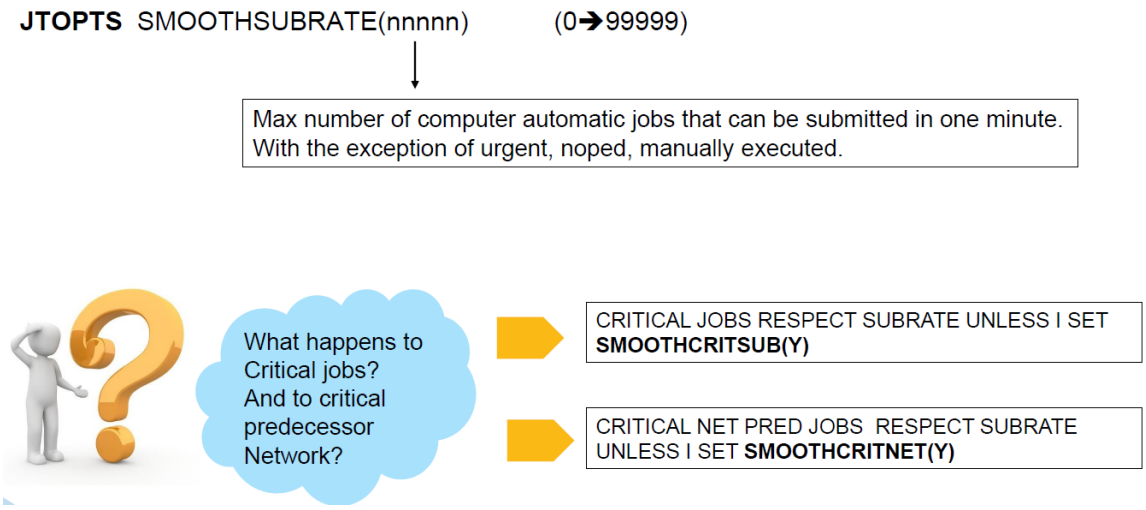
If you also want to prioritize the whole critical networks of the critical jobs, then you have to add the **JTOPTS** parameter **SMOOTHCRITNET(Y)**



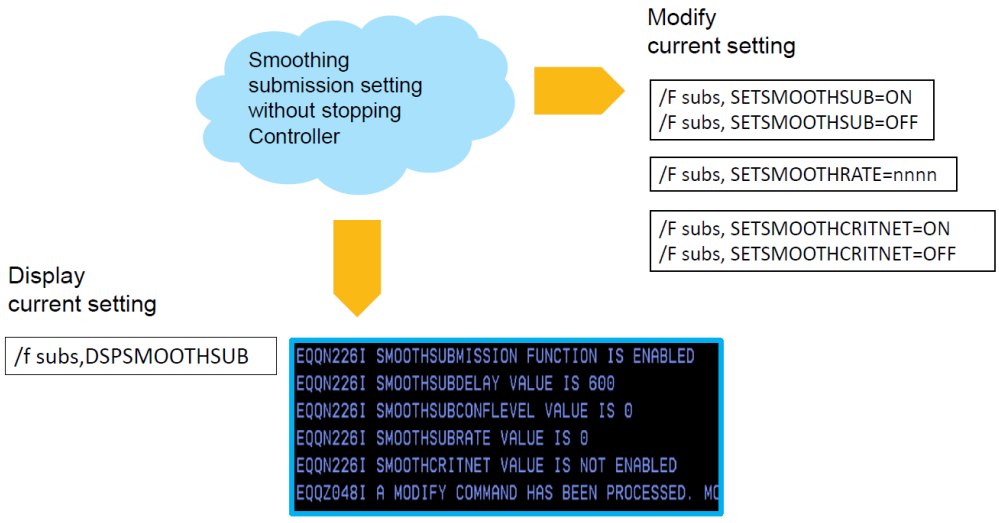
What if I am still not satisfied?

You can limit the number of submissions (of jobs) on computer automatic workstations by means of a global parameter:

JTOPTS SMOOTHSUBRATE(nnnnn) (0➔99999)



To conclude, you can also activate/deactivate/display these parameters, if you prefer:



What to do if you are in trouble with DCP?

If you see the hot lists full of entries, many critical jobs in potential (or even) in high risk, apparently without a valid reason, very likely it means one of the following:

- you made some mistakes in time settings (deadlines or estimated durations)
- your controller has been inactive for a while and when it starts a lot of operations are not started while their Latest Start Time has passed.

In both these situations the first action to do is deactivating DCP, by having the controller run with parameter CRITJOBS(NO) in the JTOPTs (be aware that the default is YES).

Once DCP is deactivated you should check your time settings, as explained in this paper, create a new plan if needed (DP batch replan or extend) to make time settings coherent and then (only then) reactivate the DCP feature (CRITJOBS(YES) or not specified in the JTOPTs).

The most frequent example of bad configuration is when for some operation you are interested in deadlines but they are not set as critical jobs. For sure this will lead to a time issue in DCP, sooner or later. To make a good use of DCP it is crucial that you adopt this definition: critical jobs are the jobs for which you are interested in the deadline. So, if you want to set the deadline for a job, please set it as a critical job.

If you did all things correctly but you are still in trouble with DCP, please contact the WSz support by providing the following documentation:

- Before running the daily plan batch, collect an **APAR tape** including the new current plan and a console dump including data spaces as shown below:

```
DUMP COMM=(reason for taking dump)
R xx,JOBNAME=(ZZZZ),CONT
R xx,DSPNAME=('ZZZZ'.*),CONT
R xx,SDATA=(COUPLE,ALLNUC,LPA,LSQA,PSA,RGN,SQA,TRT,
CSA,GRSQ,XESDATA,WLM),END
```

  where
  xx
  Specify the replay number ID.
  ZZZZ
  Specify the name of the controller.
- A complete MLOG of the controller timely compatible with the issue
- The BATCHOPTs used to create the CP.

Summary of highlights and recommendations

As I said at the beginning, using DCP when badly configured can be frustrating. However, reading a paper on configuration without a final summary of the highlights can be frustrating as well or even more.

DCP Native JTOPTS parameters

- *CRITJOBS(YES|NO)* activates/deactivates DCP. Keep in consideration that for an actual start of DCP at least a critical job must have been added to the CP via Daily Plan batch.
- *RECCPCOMPL(YES|NO)* determines if the critical path must be recalculated or not each time a job on the critical path gets completed. By default is YES, and for performance reason we recommend to set it to NO, particularly if you run with critical networks that are not very small.

DCP Native BATCHOPTS parameters

- *IGNOREDEADL(YES|NO)* moves all the deadlines of not critical jobs to the Tail End with the only exception of suppress-if-late jobs. The value YES is strongly recommended (default value is NO) to prevent false alarms in late conditions/potential risk levels when using DCP.
- *TIMEDEPCHK(YES|NO)* if set to YES, it acts in two directions:
    - Makes the Planned Start Time calculation (by DP batch) more precise. By Default, they are based on IA times even for operations that are not time dependent.
    - Allows the MCP code to initialize the Estimated Start/End Times of dynamic added workload to the current date/time rather than IA time.

Recommendations for setting Estimated Durations

The use of *Limit Feedback* and *Smoothing Factor* for durations is strongly recommended. Even one operation having a long and unrealistic estimated duration can make the use of the feature very hard, determining unrealistic latest start times calculations and, consequently, false late conditions.

The use of FIRSTFDBK can help when defining new jobs.

You can decide when your critical jobs are to be considered in high risk

RISKCONFIDENCE JTOPTS parameter allows you to set a minimum acceptable probability of matching deadline, under which the risk level becomes high.

You can try actions on the CP and see the consequences in advance

If you use DWC you have the What-If feature allowing you to see in advance the consequences of your possible actions on operations, occurrences, dependencies, workstations etc.

Pay attention to dynamic-added workload

In case you have massive dynamic-added workload and it is not made of 'stand alone' occurrences, an increase of the DP batch replans is suggested to overcome possible Estimated Time inconsistencies due to partial updates (for performance reasons).

Smoothing submissions to prioritize critical workload

When you have big amount of operations and you want to avoid system overload and job interlock you can configure WSz to prioritize the critical workload and even to determine a delay for normal workload.

You have *BATCHOPTs* parameters (SMOOTHSUBDELAY,SMOOTHSUBCONFLEVEL) to create a random delay in not critical workload. This delay will only applied if you have configured the controller to do it by means of the *JTOPTs* parameter *SMOOTHSUBMISSION(YES).* You can also decide that, together with the critical workload, also al the predecessors to the critical workload must be prioritized. In this case you use also the *JTOPTs* parameter SMOOTHCRITNET(YES).

If you are still not satisfied, you can limit the number of submission on computer automatic workstations , by using the JTOPTs parameter SMOOTHSUBRATE(99999).

.