GPM®: An Objectbase Project Networking Method
By Dr. Gui Ponce de Leon, PE, PMP, LEED AP

Abstract

CPM (Critical Path Method) applications anchored on database-driven scheduling engines controlled via keyboard and mouse have been the dominant paradigm in project management. Database-driven modeling of what is arguably a graphical application is akin to CAD - on life support until 3D Modeling/BIM pulls the plug. While CPM software vendors are focused on tricked-out scheduling engines, graphical user interfaces anchored on objectbase[1] principles and gestural computing are progressing to where they are ignored at the user's risk.

Historical Context

Owing to the technology of the times, early CPM practice was graphical and planning-centric. A "CPM" was a graphically-depicted project network (whether hand-drawn or computer-drawn) conveying activities and logic, ergo the "plan." Dates and floats were simply the mathematical corollary of the network graph. The motto was: *logic rules, dates serve*. Scheduling was an above-board exercise; what with the network model for all to see. Manipulation of a schedule was not a consideration – else the manipulator might expose himself to ridicule.

Starting in the mid-1980's, with the computational power of the PC, the practice of CPM morphed into a data-driven paradigm - sans the graphics. The power of the graphics was supplanted by the sophistication of the scheduling engine. Without a graph for all to see, eventually it became: *dates rule, logic serves*. The end game was to get the dates by crook & hook, logic was secondary. Two generations of schedulers have grown to believe that network anomalies such as endless constraint dates, negative lags, retained logic/progress override, loose ends, activities with only start-to-start or finish-to-finish successors, et. al., were tools of the trade. A "CPM" went from a network diagram to software-driven activity and logic data, conveyed by a listing and/or a bar chart to boot!

At some point, the elders of the scheduling community cried out: *we are not going to take it any more.*[2] There being far more entrenched stakeholders (i.e., software vendors and present-day schedulers) than elders, no change was effected. Arguably, the scheduling profession remains mired in the morass in which it has been for the last two decades. Except, courtesy of new, powerful graphics interfaces, new thinking and the foundational power of the mathematics underlying network graph theory, help is on the way.

The Graphical Planning Method® (GPM)

GPM transforms conventional CPM planning and scheduling by making it an engaging, interactive, real-time process anchored in computerized, graphical, objectbase project networking methods.[3] The intent is for GPM software applications to hinge on GPM algorithms for object encapsulation and message passing. Existing CPM applications rely on inline CPM scheduling engines to process data to carry out schedule calculations in a succession of batch modes. In contrast, GPM applications rely on objects that contain embedded rules and computational algorithms that interact with one another via message passing to perform planning and scheduling functions *in real time* - in addition to interactive graphics display.

Objects within GPM applications are designed to communicate with each other to carry out network activity and logic functions, resource planning, and graphics manipulation, among others. This paradigm allows intelligence, such as duration, dates, float, resource, cost, progress status, and risk factors to be encapsulated into the objects. The ability to encapsulate planning rules and computational algorithms within the objects allows resource leveling, schedule optimization, and time/cost tradeoffs to be dealt with simultaneously with network construction. In contrast, CPM relies on separate processes (or engines) to execute planning, scheduling, resource leveling and cost calculations in separate, sequential batch modes.

With GPM applications, the concurrent processing of these protocols provides a true interactive interface. Any time the schedule status quo is altered, the GPM application, owing to its object-oriented design, is capable of spontaneous healing of infeasible (corrupt) plans based on mathematical rules without having to "punt" to a stand-by CPM engine. This healing ability in the GPM algorithms provides a more cognitively responsive environment for both project management and team collaboration among multiple users and organizations, where experimenting what-if scenarios is crucial in achieving project cost/time savings.

Originally published as a part of PMICOS 2009 Annual Conference

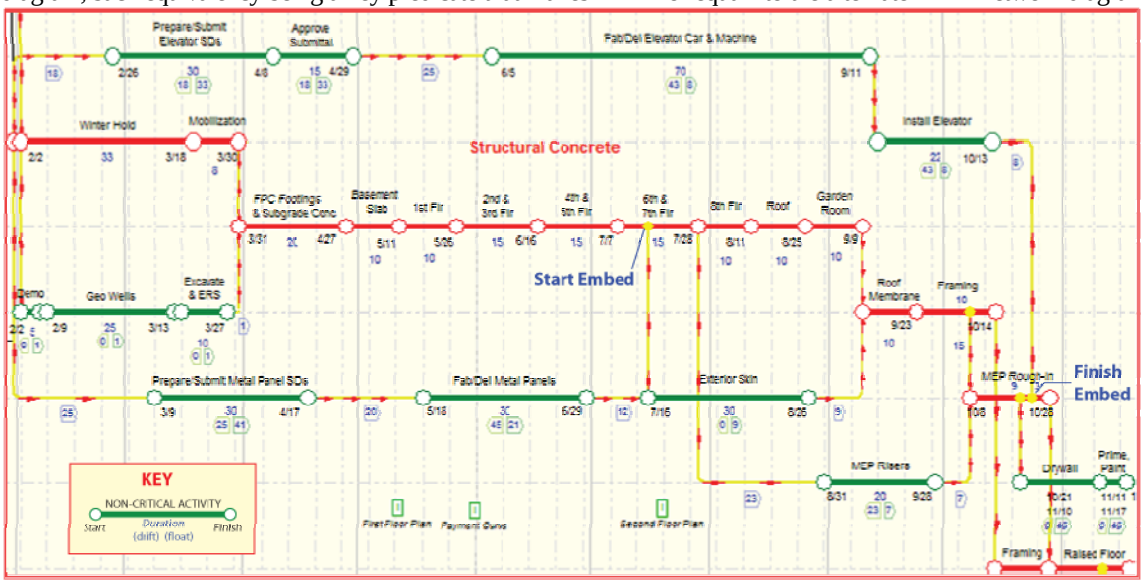
GPM-Based Planning/Scheduling Technology

Gestural and Surface Computing – CPM applications anchored in database-driven scheduling engines controlled by keyboard and mouse have been the dominant paradigm in network-based planning/scheduling for nearly three decades. Perhaps not for long – advances in computing power are enabling the emergence of hand-directed scheduling applications using touch or a stylus. Owing to its foundation on objectbase principles, where network elements are graphical objects encoded with mathematical characteristics, GPM innately supports gestural interfaces, including surface computing. Considering that the likes of Apple, HP, Microsoft and Intel are promoting gestural interfaces for the future, it is predictable that gestural scheduling applications are the future in project management as well.

Logic v Dates as the Float Predicate – The developers of CPM chose activity dates as the float predicate, to wit: total float of an activity is determined by the difference between late finish dates and early finish dates. This approach to deploying network mathematics requires that a "dates" (and "floats") calculator be on standby ready to calculate the forward pass and backward pass when planning ceases or an interim calculation is necessary for further planning. Planning and scheduling were thus bifurcated and CPM was destined to a world of batches.

Owing to its time-scaled orientation, dates are innate, real-time attributes of GPM objects. This allows GPM to use the leeway or *gap* existing in the relationship (link) between two activities, or between an activity and another GPM *event object*, as the float predicate, to wit: floats originate at the link level and can be algorithmically determined from the link *gaps* (refer to Glossary). In the GPM paradigm, activities, milestones and other event objects are encoded with the algorithms for determining floats (whether in the conventional sense of forward leeway or otherwise) from all link gaps existing across the entire network without requiring a backward pass or a separate function.

Born-Again Logic Diagram – As developed, CPM's date predicate was balanced by what came to be called the Arrow Diagramming Method (ADM), a network diagramming technique that emphasizes logic. ADM networks innately convey finish-to-start (FS) logic through the I-J notation; FS logic by its very nature avoids implied loose ends. Any activity without a predecessor (or successor) de facto has its I node (J node) untied. It can hardly be disputed that the simplicity of allowing only FS relationships eventually became ADM's undoing. By the 1990s, the alternate Precedence Diagramming Method (PDM) with its more comprehensive four types of dependencies or precedence relationships had so overwhelmed ADM that Primavera and other CPM software applications had stopped supporting ADM altogether much to the chagrin of many. Some researchers have opined that the PDM increased relationship flexibility is a double-edged sword in that, when improperly applied, it results in defective, ambiguous, hard to follow or corrupted logic.[4] For one, PDM masks loose ends, which means that any PDM activity without an FS or finish-to-finish (FF) successor in fact has its completion event unrestricted. Likewise, any PDM activity without an FS or start-to-start (SS) predecessor has its starting event unrestricted. Such inherent ambiguity in PDM techniques, however, pales in comparison to the greatest evil of network logic – negative relationship lead/lag factors. There is simply no way that a PDM network diagram using negative lead/lag factors can be converted into an equivalent ADM network diagram, such equivalency being a key predicate that makes PDM "or equal" to the alternate ADM network diagram.



Originally published as a part of PMICOS 2009 Annual Conference

On the presumption that there was no interest in the scheduling community return to the days of FS-only logic, GPM introduced the Logic Diagramming Method (LDM), a new diagramming method that combines the best of what ADM and PDM independently offer.[5] As depicted on the prior page, LDM is grounded in IJ-like notation, except that SS/FF/SF PDM dependencies can be modeled through embedded nodes intermediate of, or right on, the start and finish nodes of an LDM activity. As with ADM, LDM's time-scaled notation improves on the schematic notation favored by PDM in that implied loose or orphaned ends are readily detectable. As a matter of protocol, LDM disallows the use of negative lead/lag factors and solves the ambiguity addressed by the Relationship Diagramming Method (RDM)[6] and the orphaned ends problem and lack of clarity addressed by the Enhanced PDM Scheduling System (EPDM).[7]

Forward Pass Bias - Diagramming activities and relationships through "backward planning" techniques has been an option since the early days of CPM,[8] but over time the practice of CPM has become forward planning and forward pass-centric. The mainstream literature, except for Lean Construction and Critical Chain Project Management treatises,[9] no longer differentiates between forward and backward planning or gives priority to backward pass protocols. The forward-pass predicate in CPM is that the project start date is known from which the completion date is to be calculated. There are many projects where the completion date is a given (e.g., completion of a school in time for fall opening), and the goal of the planning and scheduling process is to assist stakeholders to determine an optimal starting date, not necessarily the earliest starting date.

GPM provides for such backward-pass activity definition, sequencing and dating, whether to entirely build a network through "pull" planning, as promoted by Lean Construction practitioners,[10] or to facilitate modeling situations where many activities converge with few successors. GPM applications are designed to competently allow planning to switch from a *push* mode (conventional forward planning) to a *pull* mode (backward planning), with predecessor activities and relationships developed backwards from already defined successor activities.

Forward Float Bias – Due to its early dates orientation, the CPM algorithm treats activity floats as measuring only slippage or delay, in other words float is thought as only measuring *forward* leeway. CPM does not calculate available schedule gain, meaning backward float, if an activity is scheduled between early and late dates or on its late dates. In contrast, the GPM algorithm treats schedule delay and gain as coexisting, subject to the following predicates:

- There is no available slippage or delay on the late dates without causing negative floats (if the project completion date is held) or delaying the project.
- There is no available earlier start or schedule gain on the early dates without causing negative float (due to violating logic) or forcing an earlier project start date.
- There is both gain and delay available if an activity is scheduled between its early and late dates.

To enable this construct, the GPM algorithm calculates activity *float* (v. total float) and activity *drift*, where for any activity, the sum value of float plus drift is a constant, and that constant equals classic CPM total float for the activity.

In GPM math, float measures available delay from current activity dates without extending completion. Drift measures available schedule gain from current activity dates without forcing an earlier start date for the project. Knowing the extent of both forward and backward flexibility for a subset of activities or the entire network provides a powerful cognitively responsive environment for team collaboration among multiple stakeholders, where knowing both floats and drifts is crucial in what-if scenarios to achieve early completion or balance resource requirements and availability.

Relationship Lead/Lag Modeling Rules – As first conceived, the lag factor predicate in PDM SS and FF relationships was intended to represent days of work, not just setoffs.[11] The rules to model logic were as follows:

- PDM SS logic – the lag models the minimum extent of progress on the predecessor activity, as measured by elapsed days of work, before the successor activity may start.
- PDM FF logic – the lag models the minimum amount of work left on the successor activity, as measured by remaining days of work, after completion of the predecessor activity.

Over time these fundamental modeling rules were lost to many, but not all, and were eventually ditched by CPM software applications. Plotnick's work on the Relationship Diagramming Method (RDM) is but one attempt to rectify this anomaly in current CPM/PDM practice.[12] In this respect, GPM is more akin to RDM than conventional PDM. More specifically GPM, through LDM diagramming techniques,[13] assigns logic attributes to embedded nodes or *embeds* (embeds are GPM event objects for PDM lags) in SS and FF relationships as follows:

- Proportionate – Embed offset, which is analogous to PDM lag value, is stated in terms of a percentage of the host activity duration (the host activity is the predecessor for SS logic and the successor for FF logic); if the host activity duration varies in any way, the embed adjusts proportionately, which would be equivalent to the PDM lag value recalculating automatically. A proportionate embed is intended to model the percent of the activity duration that must be completed for the successor to be allowed to start (SS logic) or percent of successor activity duration remaining after the predecessor completes (FF logic).
- Constant - Embed offset is measured in terms of elapsed days from the start of the host activity (predecessor for SS logic) and in terms of remaining days to the finish of the host activity (successor for FF logic); if the host activity duration varies in any way, the embed remains constant, which is typically the case with PDM applications. A constant embed is intended to model a fixed interval from the start of the predecessor to the embed (SS logic) or from the finish of the successor to the embed (FF logic), so in that sense constant embeds are consistent with the usual intent of the relationship type.
- Reverse - Embed offset is measured in terms of elapsed days to the finish of the host activity (predecessor for SS logic) and in terms of days from the start of the host activity (successor for FF logic); if the host activity duration varies in any way, the embed remains constant. For any host activity at issue, a reverse embed is the complement of a constant embed.

Forensic Float – The CPM predicate of calculating total floats from activity dates breaks down for completed activities because the late finish and early finish dates are both one and the same, more precisely, the actual finish date. Calculating total float of a completed activity as zero would misleadingly characterize a completed activity as critical and render useless an activity list with completed, in-progress and remaining activities sorted by ascending total float. The workaround, as implemented by this author in 1970's CPM software, is to supersede the total float calculation by assigning a rather large value to the float of completed activities (both for total float and free float). The downside of "turning off" float calculations for activities with actual dates is that CPM cannot calculate floats for as-built schedules nor in fact to the left of the data date for any schedule update partway through the project.

The commonly held understanding that floats ***and*** as-built schedules are not compatible in CPM scheduling is well illustrated by one of the underlying fundamentals and general principles upon which the AACEi Recommended Practice for Forensic Schedule Analysis, commonly known as RP-29, is stated to be grounded.[14]

Owing to its logic predicate for floats and drifts, GPM is not in any way stymied by actual dates when it comes to determining forensic floats or the as-built critical path for that matter. Unlike dates, any logic tie retains a real gap value even if both predecessor and successor activities are actualized, such forensic gap value no longer measuring available slippage for the predecessor, but rather whether or not the logic tie was a driving relationship in the as-built condition. The GPM float algorithm relies solely on existing gaps across the entire network, albeit on an algorithmic pattern, therefore the calculation is not affected by whether the gaps used in the equation reflect early dates, planned dates or actual dates. Unlike CPM, a GPM schedule will competently calculate forensic float and therefore competently determine the as-built critical path, whether forward (future) or behind (past side of) the data date line, including when the data date line reaches actual project completion.

Milestones – The classic understanding of milestones is that of key events or instant points in time. Unlike activities, milestones have no duration and neither consumes time nor resources. No less of an authority than the PMBOK Guide dilutes the meaning of a milestone when it states "Sometimes called a milestone *activity*," whether of its own volition or to align with prevailing CPM software thought.[15]

GPM events are objects that take place at the start of the day (restraining the start of follow-on work) or the end of the day (signaling completion of a key deliverable). If an event is fixed and not allowed to shift from its stipulated date, it is considered a *Benchmark* having zero float and drift values, whether or not logically tied to predecessors and/or successors. If the date of an event otherwise logically tied to predecessors and/or successors is determined from its predecessor or successor dates, such tracking or floating event is considered a *Milestone*. In that respect milestone events behave like zero-duration activities. As a general rule, milestones do not reduce or limit activity floats or drifts, whereas benchmarks may. The rules applying to GPM event objects are as follows:

- Start or Finish Benchmark – The date is specified by the user and not subject to calculation by the GPM algorithms. Benchmarks act as float & drift governors; if an activity with 30 days of float is tied into a benchmark on a 10-day gap link, its float is *cut* from 30 to 10 days. Similarly, if an activity with 40 days of drift is connected as a successor to a benchmark on a 5-day gap link, its drift is *cut* from 40 to 5 days.

- Finish Milestone – If logically tied to predecessors, the GPM algorithms assign that finish milestone the earliest possible date based on predecessor dates and relationship types; absent an imposed No Earlier Than date (NET), at least one predecessor would be connected to the milestone through a zero-gap link.
- Start Milestone – If logically tied to successors, the GPM algorithms assign that start milestone the latest possible date based on successor dates and relationship types; absent an imposed No Later Than date (NLT), the start milestone would be connected through a zero-gap link to at least one successor.

Resource Leveling – Perhaps the one predicate in CPM recommended practice that has survived the test of time, for better or for worse, is the fundamental tenet that in establishing activities and logic, the planner should assume that the needed resources are available to perform the network as diagrammed. That this predicate was unrealistic and impractical was not lost to the developers of CPM as Kelley and Walker in their landmark paper in 1959 provided ground-rules to adjust early dates to attain a more leveled resource profile or, if necessary, the late dates to constrain the schedule by the available resources.[16]

Owing to its strong planning orientation, GPM's resource predicate is the opposite of CPM's: one need not assume unlimited resources; rather, resources may be subject to leveling while activities and logic relationships are established. The breakthrough of not waiting for early and late schedules to consider resource leveling makes all the modeling rules available for network development also available for resource leveling. Making activity and logic decisions while simultaneously optimizing resource profiles is an efficient process in GPM. Consider the following scheme of thought by way of recommended sequence of steps and alternatives worth considering when leveling a resource within stated limits using GPM protocols:

- Push/slide or pull/backslide other activities not consuming the resource to be leveled to increase as much as possible gaps for activities consuming the resource being leveled.
- Slide (backslide) activities consuming the resource being leveled that have only positive gaps to their respective successors (predecessors) to improve the histogram or eliminate a resource limit overrun.
- Slide (backslide) activities consuming the resource being leveled that have positive float (drift) - even if connected to successors (from predecessors) via zero gap links (GPM algorithms competently cause the effected fragnets originating from or ending on the activity at issue to shift accordingly).
- Split activities with buffer and share the buffer (refer to Glossary) between segments as advantageous; slide/backslide the segments to improve the resource profile or eliminate a resource limit overrun.
- Crash or extend activities to improve the resource profile or eliminate a resource limit overrun.
- Restate finish-to-start logic, where appropriate as start-to-start or finish-to-finish logic, thereby increasing gaps and creating additional leveling alternatives.
- Crash or extend other activities not consuming the resource being leveled thereby increasing gaps and creating additional leveling alternatives.

Schedule/Cost optimization – CPM when introduced included an algorithm for shortening/extending a schedule that made it a time/cost tradeoff decision support system.[17] From 1957 to the early 1960's there were a plethora of treatises on time/cost tradeoff, most notably the work of Stanford's John Fondahl.[18] To paraphrase the late Professor Fondahl, CPM offered a systematic approach on which to base time/cost tradeoff decisions. O'Brien's scheduling handbook circa 1969 devoted 24 pages to this CPM feature under "Cost Schedule Optimization." The chapter in question leads of with "The original CPM programming included a cost optimization feature."[19]

Since then cost/schedule optimization has fallen off the deep end in CPM practice. O'Brien and Plotnick's most recent CPM text devotes only 3 pages to the subject under the heading of "Minimum Cost Expediting."[20] They posit several reasons why the optimization feature in CPM has seen little use, the last of which is the lack of available software. The Project Management Institute Practice Standard for Scheduling includes the grand total of one sentence on this CPM feature: "The most often pursued alteration involves actions to reduce the overall duration of the schedule, the key techniques used to compress the schedule are 'crashing' and fast-tracking."[21]

Owing to its foundation on network graph topological relationships and the inherent algorithms, GPM possesses all the functionality required to cost effectively shorten and extend a schedule. The GPM schedule optimization algorithm is gap-centric in the sense that it employs all link gaps for assessing the effect on the entire network of variations in an activity or group of activities targeted for crashing, extending, sliding or backsliding for the

purpose of earlier or extended completion. Due to their reliance on relationship conditions, i.e. existing gaps, the GPM optimization protocols represent and advancement on the approach developed by Professor Fondahl. The more noteworthy advancements relate to variations in an activity targeted for shortening, sliding, et al. not being bound by existing link gap values, however, any time any logic tie is corrupted by the introduction of negative gap(s), the breach is immediately healed by restoring all gaps, floats and drifts to their correct mathematical state.

All the math wizardry notwithstanding, a key benefit of the GPM schedule optimization protocol is the capability of networking occurring graphically in real-time both as to the action initiated by the user and the instantaneous, mathematically-grounded response. The resulting functionality provides a more cognitively responsive environment for team collaboration, where experimenting what-if scenarios is crucial to optimizing a schedule.

The NetPoint® GPM-Based Planning System

While the author pondered and codified the mathematical bases and algorithms of GPM, a parallel effort was engaged to develop the GPM scheme of thought into a fully functional graphics planning system. As a planning and collaboration tool, the NetPoint® System provides planning, scheduling, resource management and other project control capabilities similar to what P3/P6 from Primavera and Microsoft Project offer; however, under the hood, there exist major differences and enhancements that underscore how GPM provides a gestural, planning-centric, logic driven, collaborative, and intuitive planning environment of the future.

Conclusion

The emergence of GPM applications anchored on real-time, object-oriented graphics systems, controlled via hand touch or a stylus will inevitably change the project management landscape going forward. The power of the graphics in and of itself is compelling, particularly when contrasted with the inartful graphics available from commercially-available CPM software. The ability to witness and receive visual feedback on the impact to the schedule immediately upon any hand-directed action allows the involved stakeholders to perform at a higher level. The foundational power of the network flow algorithms underlying the method is precisely what is needed to enable an objectbase v. database-driven project networking paradigm. The functionality of GPM is closely aligned with what the likes of Apple, HP, Microsoft and Intel are predicting for the future of computing.

Glossary

Benchmark – A GPM event object designating a deadline or used to "cut" floats (drifts), if floats (drifts) are to be allocated to the phase defined by the benchmark v. accumulating from schedule completion (to project start).

Buffer – GPM-calculated activity and milestone attribute that equals the minimum of the gaps for all logic ties to successors. A GPM-perfected CPM free float concept in that it homogenizes all relationship types to a single formula.

Drift– GPM-calculated activity and milestone attribute, which measures by how many days an activity or milestone may backslide to an earlier position without forcing an earlier project start.

Embedded Node (Embed) – An event intermediate of, or right on, the start or finish node of an activity, through which the activity is connected to a successor start node, from a predecessor finish node, to/from an embed of another activity, or to a milestone or benchmark. An embed is generally offset (PDM lead/lag factor) from the finish or start node.

Float– GPM-calculated activity and milestone attribute, which measures by how many days an activity or milestone may slip beyond its position without extending overall schedule completion. An analogous concept to CPM total float, except it is measured with respect to planned dates as opposed to conventional "CPM early dates."

Gap – GPM-calculated relationship attribute measuring the leeway in a relationship: by how many days the predecessor activity may slide beyond its position or extend without delaying the successor activity's position.

Graphical Planning Method (GPM) – A graphical planning/scheduling method anchored on objectbase principles and network graph topological and mathematical rules that enables the simplest possible scheme of thought to creating and optimizing a project network schedule in the shortest possible time. At its simplest level, GPM applications support simultaneous network planning and scheduling by inexperienced professionals and even the common person.

Milestone – A GPM event object dating a key point in the schedule, which, if constrained, distributes the overall duration of the project to preset stages, and, if unconstrained, reflects the current forecasted date for the milestone.

Offsets – Required, minimum interval between the connected dates of two interconnected activities, e.g., an FS offset denotes the interval between the finish of the predecessor and start of the successor; an SS offset denotes the interval between the start of the predecessor and start of the successor; and so forth for FF and SF relationships.

Notes

1. Objectbase - a collection of rule-encapsulating objects that, in response to external events, inherently interact via message passing to drive a visual model

2. Engineering News Record (2003). *Critics Can't Find the Logic in Many of Today's CPM Schedules.* New York, NY. "What is described as a CPM schedule sometimes isn't one at all, the four experts claim. If that claim is true, it says a lot about how personal computers have transformed scheduling and what could be in store as technology reshapes other phases of the construction process. At the meeting, the four experts lamented the state of scheduling. They say they see widespread abuses of powerful software to produce badly flawed or deliberately deceptive schedules that look good but lack mathematical coherence or common sense about the way the industry works. The result is confusion, delayed projects and lawsuits."

3. Ponce de Leon, Gui (2008). *Graphical Planning Method (A New Network-based Planning/Scheduling Paradigm).* PMICOS 5th Annual Conference, Chicago, IL. Graphical Planning Method (GPM) is defined as an activity-based networking technique enabling the simplest possible scheme to connect activities and create a network schedule in the shortest possible time. It is further posited that "GPM fundamentally alters CPM by allowing evolving dates, floats, resource profiles and crash-cost curves to impact activity definition and sequencing."

4. O'Brien, James J. & Plotnick, Fredrick L. (2006). *CPM in Construction Management,* (6th Ed). New York, NY:McGraw-Hill. "PDM…is so powerful that its inner workings are rarely understood by the user. One of the most cited advantages of PDM is its use of lead and lag factors, or more succinctly, duration between activities and portions thereof to supplement the information given by the duration of the activity. Unfortunately, there does not exist a universally agreed set of definitions relating to what is meant by lead and lag factors. One result of this lack is the very software vendors may each use a differing definition without even realizing the problem."

5. Ponce de Leon, Gui (2008). *Project Planning using Logic Diagramming Method.* AACE International 52nd Annual Conference, Toronto, CA. "LDM, with its *embedded node* construct and time-scaled v. schematic diagramming synergizes what PDM and ADM individually provide. LDM's modeling of PDM logic through embeds treats PDM leads/lags as elapsed days of work related to the predecessor and successor, respectively….Further, if several activities are connected by controlling FS relationships, LDM allows common nodes to convey the relationships rather than adding unnecessary logic ties (as PDM does). This relatively simple logic construct yields a big payoff in that a chain of activities, all FS- related, is graphed as a continuous end-to-end chain rather than in a cascading, bar chart, which is the case with time-scaled PDM networks and has been the concern of researchers seeking to ameliorate their complex appearance. This is relevant in construction as construction schedules predominantly contain FS over SS/FF/SF logic (nearly 80% of the relationships is not unusual)."

6. Other researchers have independently allowed for events as opposed to embedded nodes to be placed within an activity but the posited solutions for working around the ADM and PDM hurdles lead to substantially different methods. Plotnick, Fredrick L. (2006, June). *RDM – Relationship Diagramming Method.* 2006 AACE International Annual Meeting, Las Vegas, NV.

7. Herold, Scott C. (2004, April). *Enhanced PDM Scheduling Systems.* PMICOS 1st Annual Conference, Montreal, Quebec, Canada. Addresses the lack of clarity in CPM/PDM plotted schedules, however, the recommended protocols fail to ameliorate the complexity of PDM plots. "Once the relationships are converted to activities (where the duration is equal to the relationship lag) the PDM network essentially becomes, from the computer's perspective, an arrow diagramming method (ADM) network comprised of a mixture of activities and relationships, both 'on-arrow.' The EPDM scheduling software can now analyze this network because there are no missing links between the activities. All nodes (activities and relationships) have early dates, late dates, and total float. The EPDM scheduling system has the PDM benefits of simplicity and comprehensibility, and the ADM benefits of network traceability."

8. Archibald, Russell D. & Villoria, Richard L. (1968). *Network Based Management Systems (PERT/CPM).* New York, NY: John Wiley and Sons, Inc. Authors note that the alternative to forward planning is backward planning. In backward planning, thinking and selection are reversed: what activities must be completed

before this event occurs. This eliminates the more subjective decision about what occurs next encountered in forward planning, and substitutes the more objective question of what must have already occurred for the objective event to occur. This approach will usually result in an initial network with fewer subjective biases both about dependency relationships and opportunities for concurrency.

9. Ballard, Glen and Howell, Greg (2003, July). *An Update on Last Planner*. 11th Conference of International Group for Lean Construction, Blacksburg, Virginia, US. "In summary, phase scheduling is proposed as a technique for developing a plan for completing work within a phase of a master schedule…The plans are produced using a team approach, backward pass and public allocation of schedule contingency to absorb or buffer remaining variability." The process relies on team planning and using 'yellow stickies' on a wall working backward from the completion date and incorporating interim milestones. For a backward pass application, refer to Newbold, Robert C (1998). *Project Management in the Fast Lane, Applying the Theory of Constraints*. Boca Raton, FL:The St. Lucy Press/APICS Series on Constraint Management.

10. Koskela, Lauri (2004). Moving on – Beyond Lean Thinking. *Lean Construction Journal*, 9(1) 24-37. It is noted that a production control system can be a mixed push-pull system. And that either push and pull method may be appropriate depending on the stage in question. This is relevant to construction planning in that deliveries allow the release of work, which suggests that work dependent on deliveries may best be sequenced using "push techniques. Also, see Ballard, Glen (2000). *The Last Planner System of Production Control*. PhD Thesis, University of Birmingham, Birmingham, U.K.

11. IBM, Data Processing Division (1965). *Construction Project Management Control System at the H.B. Zachry Company*. Also, IBM (1965). *1140 Project Control System: User's Manual:*

12. Plotnick, Fredrick L, Esquire, PE, (2008). *Update on RDM*. The Measurable News, The magazine of the Project Management's Institute College of Performance Management, Newtown square, PA.

13. Refer to Note 4.

14. AACE International (2007). *AACE International recommended Practice No. 29R-03, Forensic Schedule Analysis*, Morgantown, WV. "The CPM schedule used for the calculation must employ the concept of the data date. Note that the critical path and float can be computed *only* for the portion of the schedule forward (future) of the data date." *(emphasis is by this author)*. Also, **"**The as-built critical path *cannot be determined* by conventional CPM calculation alone. The as-built portion is behind (past side of) the data line, which is prior in time to the point from which CPM calculations are performed." *(emphasis is by this author)*.

15. For some unknown reason, some CPM software applications have denied milestones their event status and downgraded them to that of zero-duration activities. Such surrogate modeling creates the following problems: 1) Whether necessary or not, CPM milestones have to be connected to predecessors and successors, else they would give rise to loose ends in the sense of activities with untied beginnings or ends; 2) If a CPM milestone is to denote a contractual deadline or otherwise is not allowed to float, it becomes necessary to stipulate both NET and NLT dates to accomplish such goal.

16. Kelley, J.E. & Walker, M.R. (1959). *Critical Path Planning and Scheduling.* Proceedings of the Eastern Joint Computer Conference.

17. Kelley, J.E. (1961), *Critical Path Planning and Scheduling, Mathematical Basis.* Operations Research, Vol. 9, pp. 296-320.

18. Fondahl, John W. (1961). A *Non-computer Approach to the Critical Path Method for the Construction Industry*. Department of Civil Engineering, Stanford University, Palo Alto, CA.

19. O'Brien, James J. (1969). *Scheduling Handbook*, New York, NY: McGraw-Hill. Chapter 8.

20. O'Brien, James J. & Plotnick, Fredrick L. (2006). *CPM in Construction Management*, (6th Ed). New York, NY: McGraw-Hill. In their most recent text, O'Brien & Plotnick posit four reasons why the optimization featured in CPM has not been widely used, the last of which is the lack of available software.

21. Project Management Institute (2007). *Practice Standard for Scheduling*. Newton Square, PA.