4.2.5 Die Lösung

Wo ließe sich freilich noch ein Objekt sparen? Alle 6 standen zuvor doch auf meiner eigenen Wunschliste und sie sehen alle gemäß der Problembeschreibung aus.

„Thema", ganz klar – das ist ein „Gedankengebäude" und sowas muss nicht unbedingt programmiert werden. Auch zeigen die aktuellen Attribute von „Thema" nicht an, dass es von funktioneller Bedeutung wäre.

Wieso?

Weil es neben „Szenario" nur noch die beiden Begriffe „Input" und „Output" enthält, wie es unsere Beschrei-

bung nahelegte.

Nun sind Objekte Handlungsträger in Szenarien und je höher sie in der Hierarchie angesiedelt sind, umso weiter entfernt sind sie im Allgemeinen von direkten Eingaben, vom „Input“, umso mehr dienen sie der internen Regelung und Kontrolle. Sicher, für „Output“ liegt das anders, doch benutzten wir diese beiden Begriffe immer parallel, bei „Bedarfen“ und „Verwendungen“, bei „Impuls“, auch bei „Threads“ haben wir Eingänge und Ausgänge gleichzeitig bedacht. Wenn wir sie aber immer gemeinsam erwähnen, gehören sie auch in unserem Problembereich zusammen, und dann haben sie nichts auf der höchsten Ebene verloren. Verliert aber „Thema“ diese beiden Bedarfe, wird es ineffektiv - unser „eliminierbares“ Objekt ist damit gefunden.

Was aber tun mit „Input“ und „Output“? Zu „Impuls“ oder zu „Thread“?

Keine Idee?

Dann sehen wir doch einfach, wie sich beide Änderungen auswirken.

Zuerst ordnen wir „Input“ und „Output“ bei „Impuls“ zu und entfernen danach den Impuls, der „Szenario“ als Bedarf von „Thema“ bestimmt. Damit verschwindet „Thema“ völlig, die Anzahl verwendeter Begriff sinkt auf 18 mit einem angestrebten Szenariengewicht von 918.

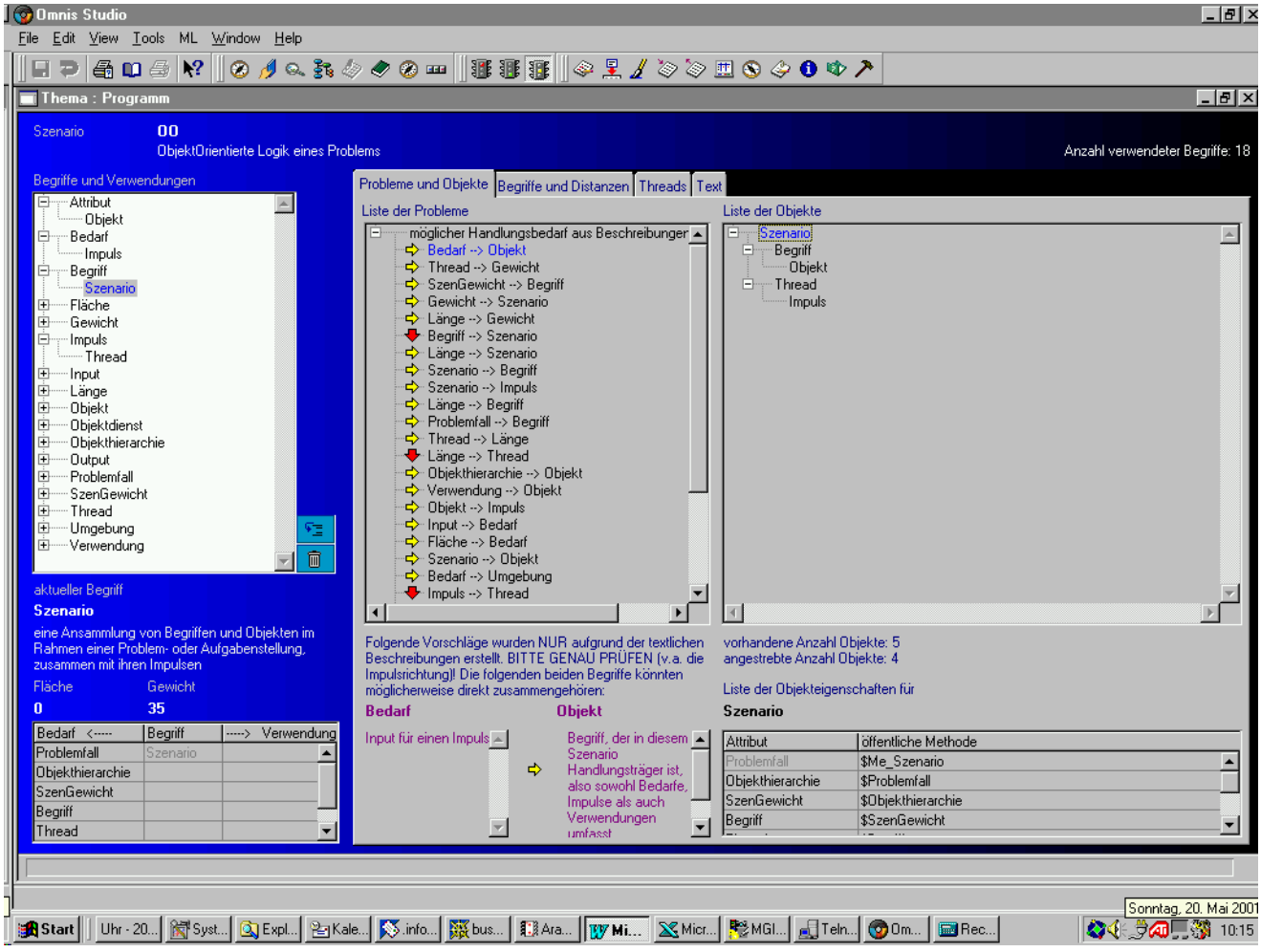

Abbildung 33 Objektliste der Lösung

5 Objekte verbleiben, in einer wirklich vernünftig aussehenden Hierarchie. Das Szenariengewicht dieser Variante liegt bei 970.

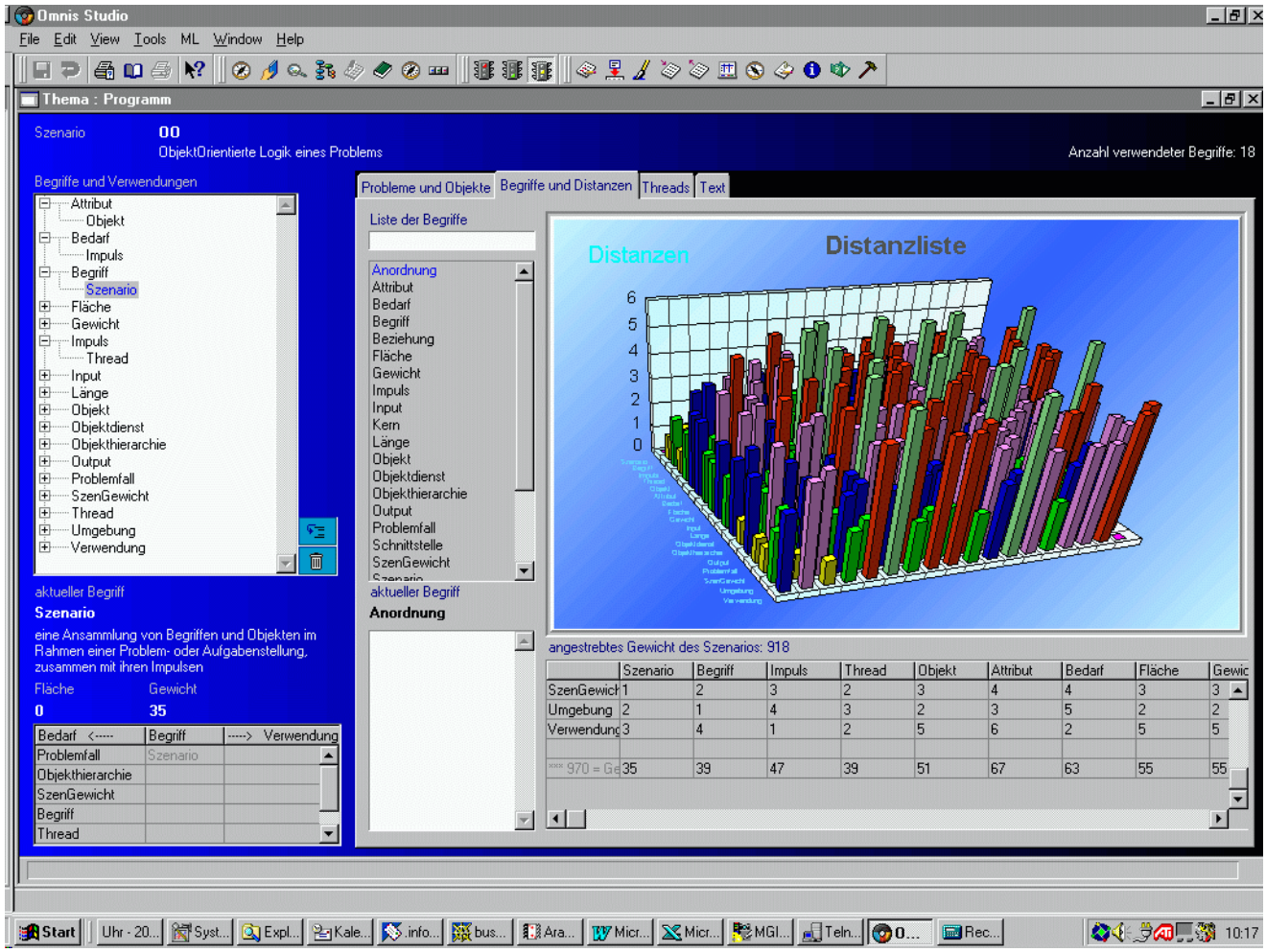

Abbildung 34 Distanzliste der Fast-Lösung

Die nächste Variation war, diese beiden Begriffe „Thread“ zuzuordnen. Das ändert die Objektliste nicht, das Szenariengewicht ist aber auf 930 gesunken. 930 liegt näher bei 918, also nehmen wir angesichts der eigenen Unentschlossenheit diese Variante, zu finden in ML_Lösung_p_ML.df1.

Im Nachhinein erscheint diese Version auch naheliegend: denn der Inhalt - Input/Output - wurde für „Impuls“ ja bereits durch „Bedarf“ und „Verwendung“ behandelt, während er bei „Thread“ noch ungeklärt war.

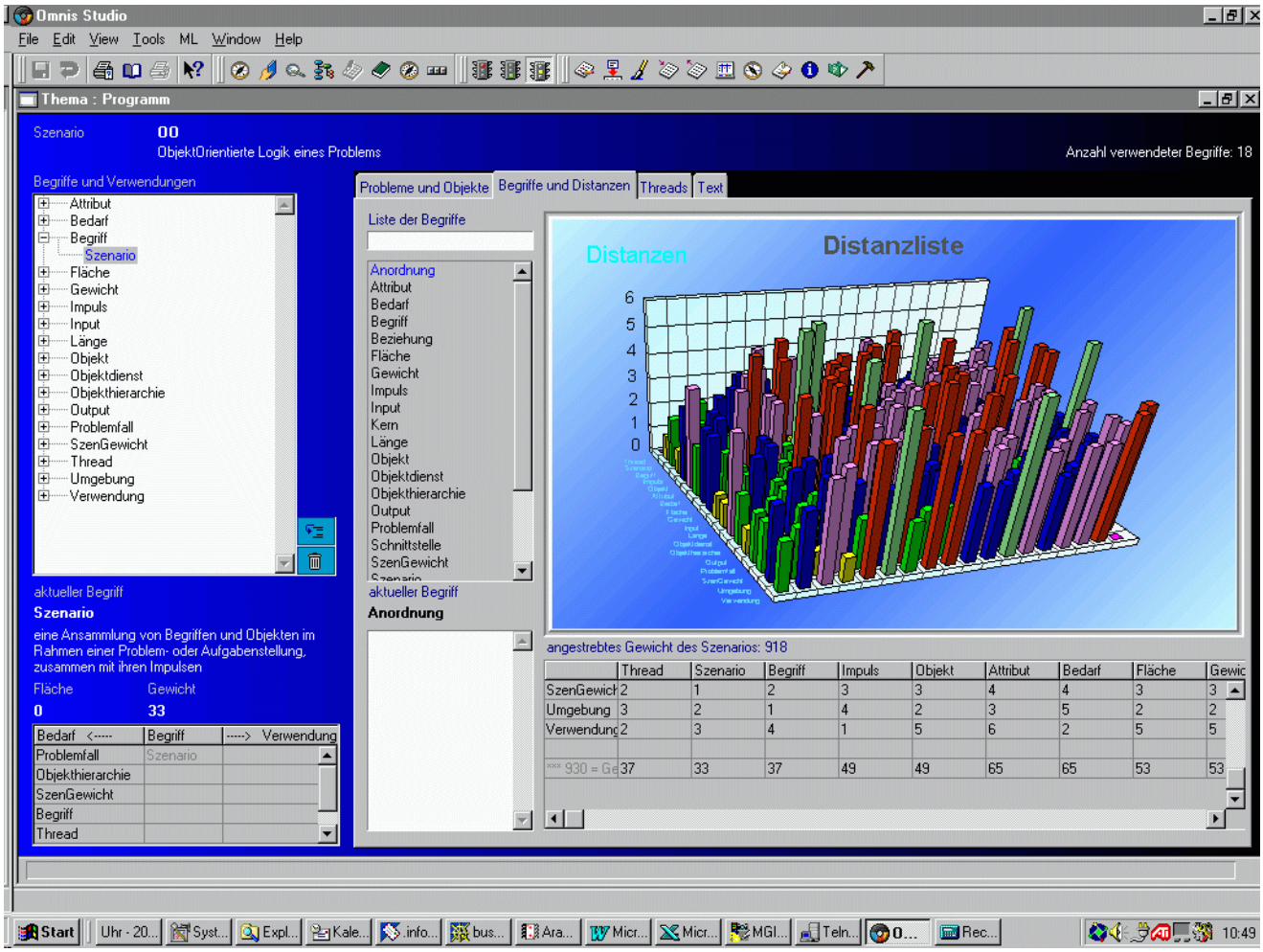

Abbildung 35 Distanzliste der Lösung

Damit haben wir die Objektliste und ihre öffentlichen Methoden und Variablen bestimmt. Sicher, das ist erst der Anfang, doch ein Anfang, der Rahmen steckt und Übersicht verschafft.

Nun müssen die Werte der Variablen abgesteckt werden, die Art und Weise, wie sie verändert werden dürfen, muss in den Methoden niedergelegt werden, es müssen Berechtigungskonzepte und Hardware-Konfigurationen berücksichtigt werden, Anbindungen an andere, existierende Software-Systeme, der ganze Alltag der Programmierung eben. Aber das ist nicht mehr Bestandteil dieses Buches, dafür gibt es eine Heerschar guter und ausführlicher Lektüre für die verschiedenen Algorithmen, Hardware- und Software-Bausteine.

4.2.6 Das Tool – Vergleich von Analyse und Realisierung

Ganz interessant aber mag sein, dass und vor allem wie dieses Tool dieser Lösung entspricht.

Das Objekt „Szenario“ ist realisiert in der relationalen Datei d_Szenario mit ihren individuellen Methoden, wobei die meisten jedoch in der Task „ML“ untergebracht sind wegen der Lesbarkeit.

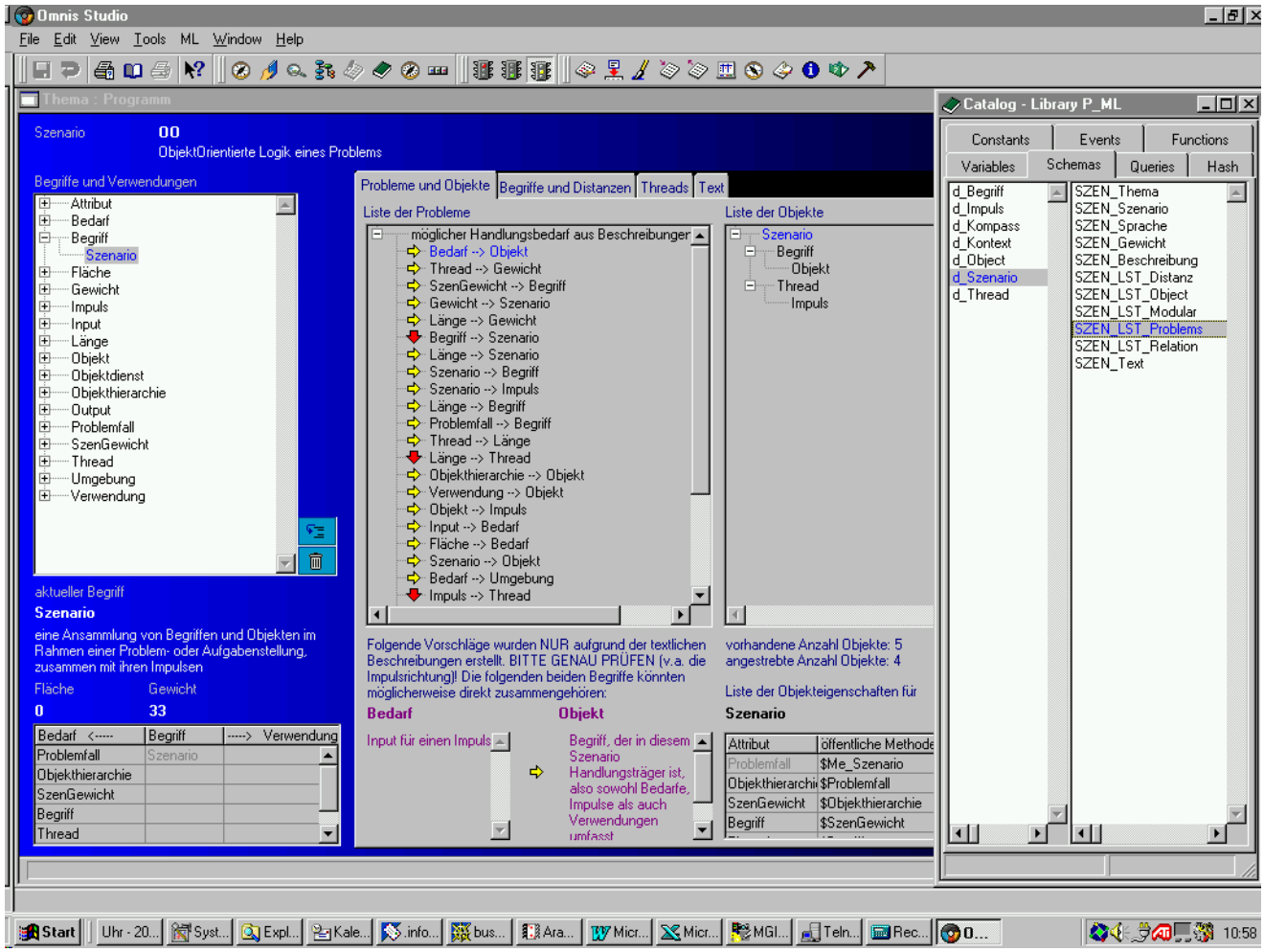

Abbildung 36 Objekt „Szenario“ der Lösung und im Tool als d_Szenario

„Thema“ hat als Ordnungsbegriff SZEN_Thema überlebt, jedoch ansonsten keine weitere Bewandtnis. „Szenario“ ist der Identifikator SZEN_Szenario und SZEN_Beschreibung als dessen verbale Darstellung. SZEN_Sprache ist nur prophylaktisch vorgesehen. „Szenariengewicht“ ist in SZEN_Gewicht niedergelegt, für dessen Berechnung auch SZEN_LST_Distanz gespeichert wurde. Dies enthält die Distanzenliste, die in unserer Analyse gar nicht besonders erwähnt worden ist. „Objekthierarchie“ ist zu SZEN_LST_Object geworden, „Problemfall“ zu SZEN_LST_Problems. SZEN_LST_Relation und SZEN_LST_Modular haben sich ebenfalls im Laufe der Detail-Programmierung als nützlich erwiesen. Sie sind zwar jederzeit errechenbar aus den Vorgaben, doch da die Rechnerei zeitaufwändig ist, wurde der Weg der Speicherung des Rechenvorgangs gewählt. Dies ist zwar nach den Regeln der reinen Datenbank-Lehre als Redundanz eigentlich verwerflich, doch da die Individualität in den Daten liegt, sollte dort auch hinterlegt werden, was nicht nur nicht wiederbeschafft werden kann, sondern eben auch, was nur sehr umständlich wiederbeschafft werden kann – oder was als Nachweis zu dienen hat, wie beispielsweise in Software, die mit juristisch heiklen Themen oder mit Geld umgehen muss. Die allgemeine verbale Beschreibung unseres Szenarios, die in der Analyse ebenfalls hinter dem Begriff „Szenario“ steckt, ist im Feld SZEN_Text zu finden.

Relationales Dateisystem und Komponentenobjekte

Warum „Thread“ und „Begriff“ nicht auftauchen? Weil wir ein relationales Dateisystem verwenden, in dem Komponentenobjekte nicht bei dem übergeordneten Objekt vorliegen, sondern durch Schlüsselfelder zugeordnet

werden. Sowohl in der Datei d_Thread als auch in d_Kontext, was dem Begriff „Begriff“ entspricht, liegen somit die Felder „Thema“ und „Szenario“ vor, um diese Bezüge zu demonstrieren

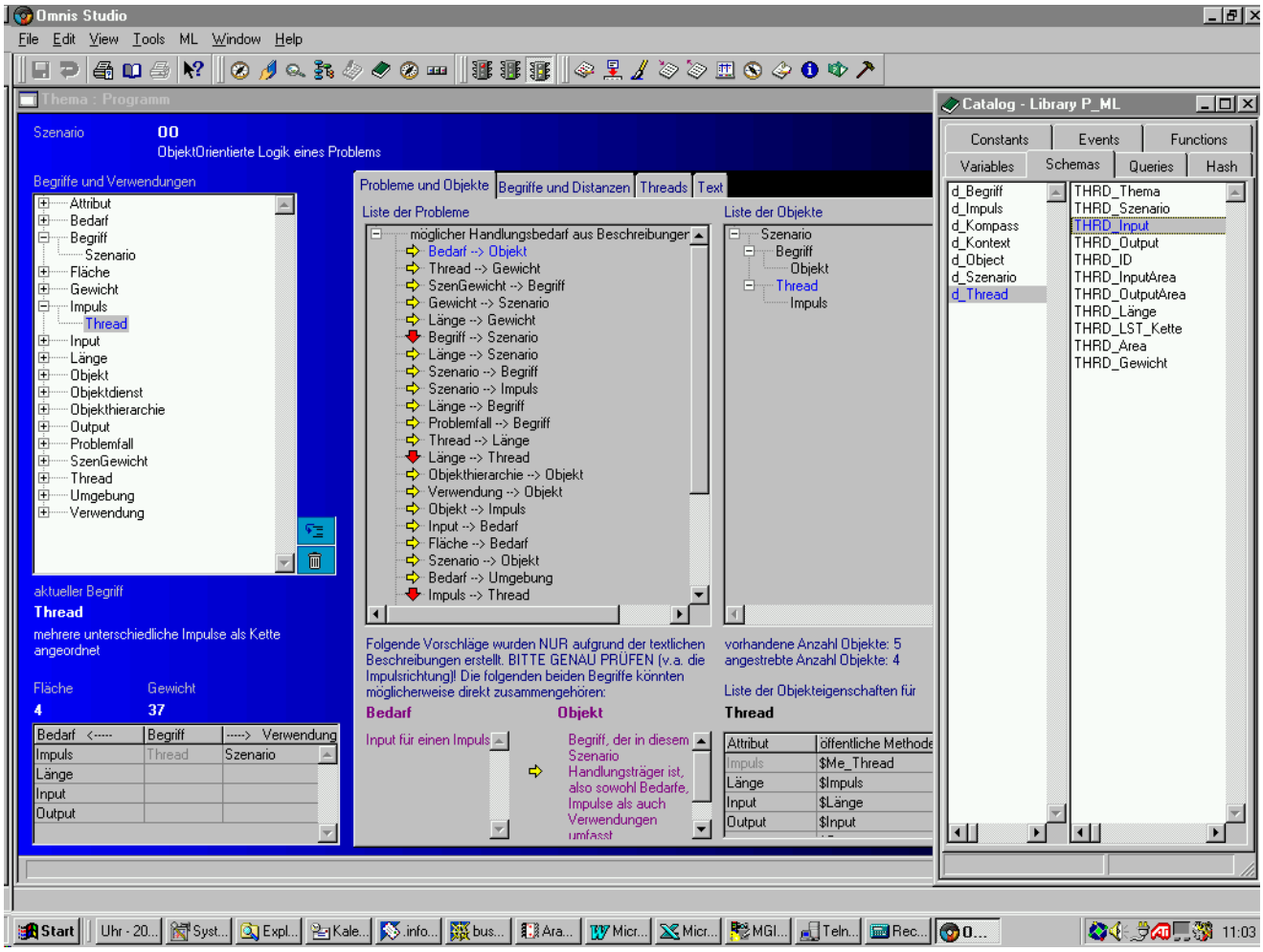

Abbildung 37 Objekt „Thread“ der Lösung und im Tool als d_Thread

d_Thread bildet das Objekt „Thread“ dann ganz ähnlich ab. Die Zuordnung zu „Szenario“ und „Thema“ ist über Schlüsselfelder geregelt, die Felder THRD_Input und THRD_Output bezeichnen die beiden Objekte oder Variablen, die am Anfang und am Ende der Impulskette stehen, die hier in THRD_LST_Kette vermerkt ist. In dieser Kette ist der „Impuls“ verborgen, zwar nicht durch eine eindeutige, über Schlüsselfelder regelbare, sondern nur über eine mehrdeutige, aber immer noch klare Beziehung. THRD_ID ist nur ein Laufzähler, der während der Detail-Ausarbeitung auftauchte, da Threads zwischen zwei Begriffen sehr wohl in unterschiedlichen Variationen vorkommen können, weshalb dieses Feld der Identifikation des Threads zuzuordnen ist. Die Felder für THRD_InputArea und THRD_OutputArea sowie THRD_Area und THRD_Gewicht sind nur zur Information, sie zeigen die Begriffe und die entsprechenden Werte in der Impulskette an, die gewichts- und flächenmäßig besonders hervorragen.

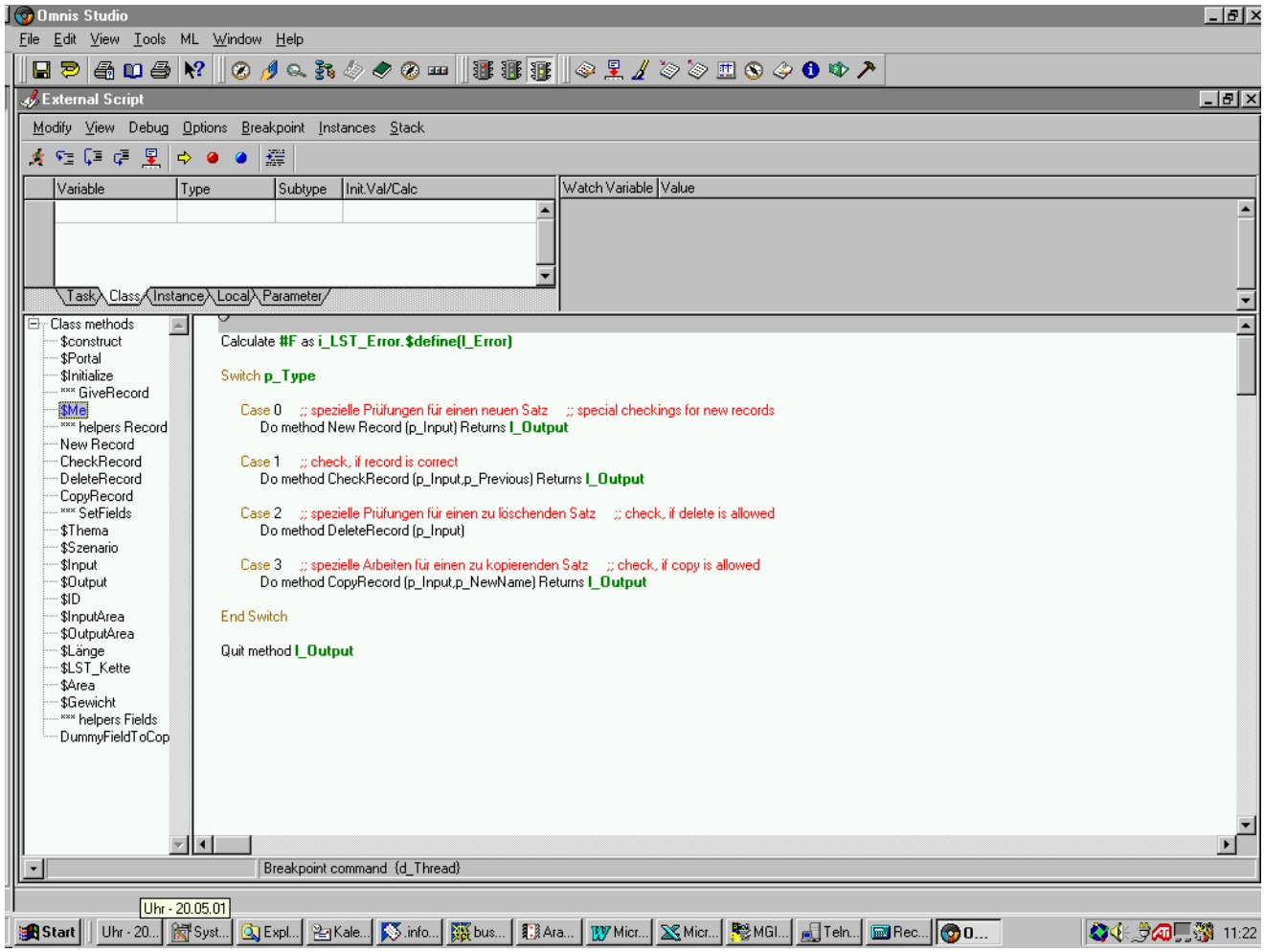

Abbildung 38 Objekt „Thread“ der Lösung und die individuellen Methoden im Tool

Im Tool wurden die Objektmethoden der Datei d_Thread als Dateifelder der Metadaten-Datei d_Kompass hinterlegt, die in den datenanonymen Dateizugriffs-Tasks t_ML mit anderen Dateiarbeiten ausgeführt werden. Ihr Aufbau ist mit Ausnahme der ersten Initialisierungsroutinen mit den öffentlichen Methoden der Analyse identisch, wobei interne Methoden und Variable zur Ausführung dieser öffentlichen Methoden völlig unberührt bleiben. Nur die Methode $Me_Thread wurde namensmäßig auf $Me reduziert.